UNIVERSIDAD POLITÉCNICA SALESIANA
SEDE QUITO

CARRERA:
INGENIERÍA DE SISTEMAS

Trabajo de titulación previo a la obtención del título:
INGENIEROS DE SISTEMAS

TEMA:

DISEÑO Y DESARROLLO DE UNA PLATAFORMA WEB PARA COMERCIALIZACIÓN DE PRODUCTOS AGRÍCOLAS DE LAS COMUNIDADES CAMPESINAS DEL ECUADOR

AUTORES:
PERRAZA SARABIA MARLON ISRAEL
TOCTAGUANO VINCES DANNY MAURICIO

TUTORA:
LINA PATRICIA ZAPATA MOLINA

Quito, agosto del 2021

CESIÓN DE DERECHOS DE AUTOR

Nosotros, Perraza Sarabia Marlon Israel con documento de identificación N° 1716685761 y Toctaguano Vinces Danny Mauricio con documento de identificación Nº 0704405927, manifestamos nuestra voluntad y cedemos a la Universidad Politécnica Salesiana la titularidad sobre los derechos patrimoniales en virtud de que somos autores del trabajo de titulación intitulado: DISEÑO Y DESARROLLO DE UNA PLATAFORMA WEB PARA COMERCIALIZACIÓN DE PRODUCTOS AGRÍCOLAS DE LAS COMUNIDADES CAMPESINAS DEL ECUADOR, mismo que ha sido desarrollado para optar por el título de: INGENIEROS DE SISTEMAS, en la Universidad Politécnica Salesiana, quedando la Universidad facultada para ejercer plenamente los derechos cedidos anteriormente.

En aplicación a lo determinado en la Ley de Propiedad Intelectual, en mi condición de autor me reservo los derechos morales de la obra antes citada. En concordancia, suscribo este documento en el momento que hago entrega del trabajo final en formato digital a la Biblioteca de la Universidad Politécnica Salesiana.

Quito, agosto del 2021

……...
Marlon Israel Perraza Sarabia
CI: 1716685761

……...
Danny Mauricio Toctaguano Vinces
CI: 0704405927

DECLARATORIA DE COAUTORÍA DEL DOCENTE TUTOR

Yo declaro que bajo mi dirección y asesoría fue desarrollado el Proyecto Técnico, DISEÑO Y DESARROLLO DE UNA PLATAFORMA WEB PARA COMERCIALIZACIÓN DE PRODUCTOS AGRÍCOLAS DE LAS COMUNIDADES CAMPESINAS DEL ECUADOR realizado por Marlon Israel Perraza Sarabia y Danny Mauricio Toctaguano Vinces, obteniendo un producto que cumple con todos los requisitos estipulados por la Universidad Politécnica Salesiana, para ser considerados como trabajo final de titulación.

Quito, agosto del 2021.

……...

LINA PATRICIA ZAPATA MOLINA
CI: 0501877278

DEDICATORIA

Quiero dedicar este proyecto y carrera a mi tío Patricio Chimbo, una de las personas que más anheló que culminara esta etapa de mi vida, por su apoyo, por sus consejos, su ejemplo y sobre todo por su tiempo compartido, desde el cielo espero se sienta orgulloso por alcanzar esta meta.

A mi tía Patricia Medina siempre estuvo al pendiente de mi vida, para llegar a ser un profesional, me guío desde niño para poder alcanzar este objetivo.

A mi amada madre Jenny Sarabia por la paciencia y el amor, por todo el sacrificio de todos esto años que paso lejos, que hoy tiene su merecida recompensa, puedo decir que pude culminar mis estudios gracias a ti.

A mi amado padre Eduardo Perraza por el apoyo brindado, por los consejos, siempre quiere lo mejor para mí y espera ver a su hijo convertido en profesional.

Para mis hermanos Alex, Verónica y María José siempre me apoyaron cuando los necesite para conseguir mi meta, siempre confiaron en mí. A Luis Guayta mi hermano, mi amigo y un padre por el cariño y el apoyo que me brinda.

A mis sobrinos Jhoel, Ambar, Camila, Celeste, Evan, Dilan y Vayoleth que este proyecto sea un mensaje para su vida, alcancen y cumplan todas sus metas.

A mi sobrino Paulito partió al cielo muy pronto y ahora está junto a su abuelo Patricio, ha cuidado de mí y sepa que lo llevo en mi corazón.

A mis abuelitos Julio Sarabia y Emperatriz Medina que con su amor y cariño siempre me apoyaron a mis difuntos abuelitos Hugo Perraza y Angelita Barrera están presentes en mi corazón siempre cuidaron de mí y me brindaron su amor, están en mi corazón.

Marlon Israel Perraza Sarabia

DEDICATORIA

Este proyecto va dedicado a mis queridos padres Zoila Vinces y Luis Toctaguano quienes siempre me han dado su cariño, sacrificio y apoyo incondicional, me han permitido culminar con mis estudios universitarios y así cumplir una meta más en mi vida.

A mis difuntos abuelos Emma Pisco y José Vinces que siempre están presentes en mi corazón, porque su recuerdo me da fuerzas y me impulsa a salir adelante, a mi abuela Delia Toctaguano por darme su cariño y apoyo a través de los años.

A mi familia quienes supieron acompañarme en este largo trayecto brindándome palabras de aliento y motivación, también a mis amigos que durante todo el proceso supieron ayudarme y el también haber compartido vivencias dentro de la vida universitaria.

Danny Mauricio Toctaguano Vinces

AGRADECIMIENTOS

En primer lugar, agradezco a Dios, por la vida, la salud y oportunidad de concluir mis estudios.

A mis padres Jenny, Eduardo, Patricia y Patricio todos fueron esenciales con su apoyo y su cariño en cada aspecto de mi vida, no me alcanzara el tiempo para pagar todo lo que me brindaron. A mis hermanos y sobrinos me inspiraron para alcanzar esta meta.

A mis tíos Edwin Cueva y Jenny Perraza me cuidaron y apoyaron como a un hijo, fueron importantes en mi vida para llegar a este momento.

A Edwin, Katty y Vayoleth por su apoyo y su cariño siempre ha creído en mí, sabían que conseguiría mi meta, nunca lo dudaron y siempre están cuando más necesito de ellos. Así como a mi prima Sylvia, también confió en que concluiría mis estudios.

A mi tío Ricardo sus consejos fueron importantes en mi etapa académica.

A Jazmín Pineda por el apoyo incondicional, la persona que me apoyo en el momento más duro de mi vida, por su paciencia, por su comprensión y sobre todo por su cariño no me dejo perder el camino y me recordó que aún existe cosas por las que luchar.

A mi amigo Jonathan por el tiempo compartido en la universidad por el apoyo que tuve de tu parte tanto dentro como fuera de la universidad. A mi amigo Danny, por el apoyo que me brindo en el proyecto de titulación y el tiempo que compartimos en la Universidad.

A nuestra tutora la ingeniera Lina Zapata mis más sinceros agradecimientos, por su disposición para el desarrollo de este proyecto hasta su culminación.

A la Universidad Politécnica Salesiana y sus docentes de la carrera de Ingeniería de Sistemas por los conocimientos y valores que han contribuido en mi formación académica.

Marlon Israel Perraza Sarabia

AGRADECIMIENTOS

A mis padres quienes han sido la motivación y el apoyo incondicional que he tenido a lo largo de mi vida tanto universitaria como personal. A la Universidad Politécnica Salesiana y a todos los docentes de la carrera de Ingeniería de Sistemas que han contribuido en mi formación académica. A nuestra tutora Ing. Lina Zapata mis más sinceros agradecimientos, quien gracias a su tutela y asesoría supo guiarme en el desarrollo de este proyecto.

Danny Mauricio Toctaguano Vinces

INDICE

ÍNDICE DE TABLAS

ÍNDICE DE FIGURAS

RESUMEN

El presente trabajo de titulación consiste en desarrollar una aplicación web para la compra y/o venta de productos agrícolas de forma online, la cual permitirá a las comunidades campesinas del Ecuador comercializar sus productos de forma directa sin intermediarios y darse a conocer de forma exponencial y así llegar a más consumidores.

Para el desarrollo del proyecto se utiliza diferentes tecnologías que permiten la creación de la página web, estas son NodeJs que permite trabajar con JavaScript en el lado del servidor, MySQL que permite la gestión de la base de datos, Express es un framework que proporciona un conjunto solido de funciones para crear aplicaciones web y API.

Una vez diseñado el sistema este proporcionara una mejor gestión de los productos que posee el emprendedor dándole herramientas para administrar los mismos y también contará con un módulo que permite la gestión de pedidos para que los clientes tengan la opción de comprar los productos agregados en el sistema.

ABSTRACT

This degree work consists of developing a web application for the purchase and/or sale of agricultural products online, which will allow the rural communities of Ecuador to market their products directly without intermediaries and to make themselves known exponentially and thus reach more consumers.

For the development of the project we use different technologies that allow the creation of the website these are NodeJs that allows working with JavaScript on the server side, MySQL that allows the management of the database, Express is a framework that provides a solid set of functions to create web applications and API.

Once the system is designed it will provide a better management of the products that the entrepreneur owns giving him tools to manage them and also will have a module that allows the management of orders so that customers have the option to buy the products added in the system.

INTRODUCCIÓN

ANTECEDENTES

En la actualidad no existe muchas opciones de plataformas web en la que productores o emprendedores de las comunidades campesinas del Ecuador, puedan promocionar, gestionar y tener un control de los datos de los productos comercializados. Con esta idea se busca dar visibilidad a cada uno de los productos que ofrecen los emprendedores, con el fin de ampliar y mejorar sus ventas. El proceso de compra y cotización de los productos por parte de los clientes como es habitual es llegando a los puestos de ventas en los mercados, si bien estos lugares o plazas han permitido a los productores tener un espacio para vender sus productos, esto no significa que siempre los mismos productores sean los que vendan directamente a los compradores, desafortunadamente los productores tienen que valerse de terceros para ofrecer sus productos, ya que estos no cuentan con un espacio o los recursos para la venta directa de sus cultivos. Por eso se ha visto la necesidad de crear un sistema web, con esta herramienta se brindará información online sobre los productos que cada productor puede ofrecer.

PROBLEMÁTICA

Hoy en día la venta y compra de productos agrícolas se ha visto afectada por la pandemia que azota a la población mundial, por lo cual muchos de ellos ven la posibilidad de integrar las Tecnologías de la Información y Comunicación (TIC) a sus emprendimientos, para gestionar la información de los productos y así lograr estar a la par con las nuevas tendencias que actualmente se llevan a cabo. Es primordial que los productores vean nuevas alternativas, que ofrezcan ventajas competitivas dándoles la facilidad de crecer exponencialmente y generando ganancias acordes a su trabajo.

El siguiente proyecto tiene como fin la creación de una plataforma web dedicada a la compra y venta de productos agrícolas, dando la posibilidad al comprador elegir diferentes productos de forma rápida sin verse afectada la calidad de la misma, además le permite tener

una relación directa con el vendedor. A sí mismo, los productos que se lleguen a comercializar, se encontrarán dentro de categorías los cuales están debidamente clasificados. Cada producto constará de una descripción, fotografía, precio, ubicación y teléfono del emprendedor, será el productor el responsable de realizar la gestión de este material, así como su correcta organización.

Se tiene como objetivo apoyar a pequeños emprendedores en la difusión y comercialización de sus productos, entregándoles herramientas de apoyo para que los productores puedan darse a conocer, en este caso con la ayuda de una plataforma web, esto con el fin de llegar de manera masiva a clientes potenciales, permitiendo que cada uno de los productores se conviertan en emprendedores plenamente competitivos.

JUSTIFICACIÓN

Actualmente los productores del Ecuador no reciben un precio justo por sus productos, ya que son los intermediarios los que generan ganancias a través de ellos y esto provoca que muchos productores busquen nuevas alternativas de promocionar sus productos que no sea de forma tradicional, buscando que sus productos lleguen a más consumidores de forma directa, teniendo como barrera plataformas web para poder promocionar sus productos.

Ante la problemática descrita proponemos la creación de una plataforma web para facilitar y automatizar la venta de productos. Las nuevas tecnologías han abierto un abanico de posibilidades y más aún en el contexto que estamos actualmente viviendo, haciendo que muchos productores no puedan salir a vender u ofertar sus productos, situación que más que verse como desventaja, viene a ser una nueva oportunidad para que los pequeños y medianos productores agrícolas crezcan en sus ventas, dando a conocer sus productos por medios tecnológicos, como la Internet, aplicando Tecnologías de Información y Comunicaciones (TICS) dando lugar a la creación de la plataforma web, esto permitirá un acercamiento directo entre agricultores(emprendedores) y consumidores para fines comerciales. Esta herramienta será de

gran apoyo a las dos partes porque permitirá la compra-venta de productos agrícolas a precios más justos y de buena calidad.

A través de la plataforma web se busca dar visibilidad difundiendo información referente a los productos que ofrece cada productor con miras a incrementar sus ventas y su cartera de clientes.

OBJETIVOS

OBJETIVO GENERAL:

Desarrollar una plataforma web para la compra-venta de productos agrícolas Online, ofertados por emprendedores de las diferentes comunidades campesinas del Ecuador.

OBJETIVOS ESPECÍFICOS:

Diseñar una aplicación web para que los pequeños agricultores registren y comercialicen sus productos agrícolas en forma directa con los consumidores finales.

Proporcionar un mecanismo de compra y/o venta en línea, a través de la Internet, productos agrícolas a precios más justos

Implementar un módulo de ofertas y descuentos para promocionar productos y empresas auspiciantes.

Entregar la aplicación web cuya funcionalidad cumpla con los requerimientos del usuario final.

MARCO METODOLÓGICO

Metodología XP

La metodología aplicada para este proyecto es la de XP, ya que brinda agilidad y flexibilidad tomando en cuenta además cuatro variables las cuales son costó, tiempo, calidad y alcance. Está metodología también conocido como programación extrema (Extreme programming) y se centra en la creación de un producto basados en los requerimientos del cliente.

Fase de planificación

El desarrollo de la plataforma web tiene como finalidad ofrecer a los emprendedores de las diferentes comunidades del Ecuador un sitio en el que puedan comercializar los productos agrícolas o emprendimientos de forma directa con los consumidores finales.

Historias de usuarios

Son las descripciones de cada una de las características, requerimientos y funcionalidades del sistema, estos van a ser escritos por el cliente en una tarjeta, en estas tarjetas se cataloga las prioridades que tiene cada característica o requerimiento, es importante recordar que la tarjeta no sobrepase las 3 semanas de desarrollo, en ese caso el cliente tiene que dividir las historias de usuario en dos historias más pequeñas y los desarrolladores realizarán el que tenga mayor grado de prioridad.

Roles

Jefe de proyecto: Es la persona que se encarga de supervisar y asegurar que se cumplan los objetivos y los tiempos de entrega de cada uno de los módulos del proyecto

Cliente: Define cada una de las características y especificaciones que va a tener el sistema.

Programadores: Encargados del desarrollo e integración de los módulos, desarrollo del código fuente y responsables del diseño y arquitectura de la plataforma web.

Tracker: Analiza y recolecta la información sobre el desarrollo del proyecto, supervisa el cumplimiento de las estimaciones y controla las pruebas funcionales.

Plan de entregas

El plan de entregas es el cronograma del proyecto, donde se establece las fechas en las que se entrega los módulos descritos en las historias de los usuarios, para esta entrega se realizará una reunión de proyecto, donde estarán presentes cada uno de los interesados y se determina la presentación de las siguientes actividades y entregables.

Plan de iteraciones

El plan de iteraciones permite dividir historias de usuarios que sobrepasen las 3 semanas de desarrollo, en este plan también se cambia la prioridad de una historia ya existente y se puede agregar o eliminar historias.

Fase de diseño

Realizar un diseño entendible, simple y de fácil uso, la recodificación que consiste en la modificación del código donde se pueda simplificar sin alterar la funcionalidad del sistema. Utilizar metáforas permite el uso de nomenclaturas para mantener coherencia tanto en los nombre, métodos, funciones y clases.

Fase de codificación

Características de la codificación:

- Programación en parejas
- Integración de código
- Uso de estándares
- Código abierto es propiedad de todos
- Disponibilidad de tiempo del usuario

Fase de pruebas

Al terminar con la integración de cada uno de los módulos del sistema se procede a las pruebas, estas fueron establecidas en la fase de planificación del proyecto.

Pruebas unitarias

Las pruebas serán aplicadas con el funcionamiento del código que se vaya desarrollando en cada uno de los módulos, esto permitirá la detección de los errores y corrección de “bugs”.

Pruebas de aceptación

Son llamadas también pruebas del cliente estas van a ser específicamente realizadas por el cliente dónde va a verificar las características y funcionalidades del sistema. Estas pruebas de aceptación se realizan partiendo de las historias de los usuarios.

CAPÍTULO 1

MARCO TEÓRICO

1.1.Aplicación web

Una aplicación web es una aplicación o herramienta informática accesible desde cualquier navegador, bien sea a través de internet (lo habitual) o bien a través de una red local. (Neosoft , 2018)

En una aplicación web los datos y la información se almacenan en BASES DE DATOS (BD). Estas están formadas por un número variable de tablas que contienen columnas y filas, estas tablas se componen del contenido que ha sido previamente cargado en ellas a través de formularios.

En estas tablas llamamos al nombre de cada columna campo y a cada fila registro, ambos en el sistema poseen códigos de identificación que son únicos para cada uno de ellos.

Las páginas que se generan a partir de esos contenidos son llamadas dinámicas. En este contexto el término dinámico no indica movimiento o animación, sino que hace referencia al hecho de que estas se generan a partir de una solicitud o consulta que realiza un usuario desde un navegador (también llamado cliente) a un servidor web. Se podría decir que la página dinámica no existe hasta que no es solicitada por el usuario. Cuando este la solícita se ejecuta una consulta a la base de datos, y el sistema muestra una página web con el contenido solicitado. (emfasi, 2014)

1.2.Sistemas gestores de base de datos

Es un conjunto de programas que permite la gestión y administración de base de datos, en la cual se define una estructura para el almacenamiento de la información evitando redundancia de datos y garantiza la calidad de los datos. Los gestores de base de datos son utilizados como una interfaz para la comunicación de los usuarios finales con las bases de datos. (deyde, 2020)

Para el siguiente proyecto se decidió usar MySQL la cual presenta algunas ventajas que lo hacen muy interesante para los desarrolladores. Una de las más importantes es que permite trabajar con bases de datos relacionales, esto permite trabajar con tablas múltiples que se relacionan entre sí para almacenar y organizar la información, cuenta con una gran comunidad que ofrece soporte a todos los usuarios. A continuación, se muestra algunas características:

Arquitectura Cliente y Servidor: MySQL se basa en el modelo cliente y servidor. Lo que indica que el cliente y servidor se comunican entre sí, pudiendo realizar consultas a través del sistema.

Compatibilidad: SQL es un lenguaje generalizado dentro de la industria. Al ser un estándar MySQL es posible la compatibilidad.

Desencadenantes: Permite automatizar ciertas tareas dentro de nuestra base de datos, los conocidos son los denominados eventos. (Robledano, 2019)

1.3.Lenguajes de programación

En términos generales, un lenguaje de programación es una herramienta que permite desarrollar software o programas para computadora. Los lenguajes de programación son empleados para diseñar e implementar programas encargados de definir y administrar el comportamiento de los dispositivos físicos y lógicos de una computadora. Lo anterior se logra mediante la creación e implementación de algoritmos de precisión que se utilizan como una forma de comunicación humana con la computadora.

A grandes rasgos, un lenguaje de programación se conforma de una serie de símbolos y reglas de sintaxis y semántica que definen la estructura principal del lenguaje y le dan un significado a sus elementos y expresiones.

La función principal de los lenguajes de programación es escribir programas que permiten la comunicación usuario-máquina. Unos programas especiales (compiladores o

intérpretes) convierten las instrucciones escritas en código fuente, en instrucciones escritas en lenguaje máquina (0 y 1).

Los intérpretes leen la instrucción línea por línea y obtienen el código máquina correspondiente.

En cuanto a los compiladores, traducen los símbolos de un lenguaje de programación a su equivalencia escrito en lenguaje máquina (proceso conocido como compilar). Por último, se obtiene un programa ejecutable. (Ceballos, 2004)

Para la creación de la aplicación web se usará como lenguaje de comunicación JavaScript la cual permite el desarrollo de sitios web, permitiendo páginas de mayor interactividad y dinamismo. Cuando JavaScript se ejecuta en el navegador, no necesita de un compilador. El navegador lee directamente el código, sin necesidad de terceros. Por tanto, se le reconoce como uno de los tres lenguajes nativos de la web junto a HTML (contenido y su estructura) y a CSS (diseño del contenido y su estructura). (Ramos, 2020)

JavaScript presenta varios Framework para el Frontend como: Angular, Vue.js, Ember.js, Backbone.js, Mercury.js, React.js y para el Backend Node.js, estos Framework ayudan a los desarrolladores ya que reducen el tiempo en el desarrollo.

1.4.Control de versiones

El control de versiones permite a los desarrolladores detectar las modificaciones y mejoras que se realizan en el código fuente a lo largo del tiempo, de esta manera si un desarrollador comete un error puede dirigirse a una versión anterior, comparar los códigos para detectar y corregir el error, de esta manera reduce el tiempo de interrupción del equipo de desarrollo. (Atlassian, 2019)

Para el siguiente proyecto se decidió usar git, ya que es una de las mejores herramientas de control de versiones disponible en el mercado actual. Es un modelo de repositorio distribuido

compatible con sistemas y protocolos existentes como HTTP, FTP, SSH y es capaz de manejar eficientemente proyectos pequeños a grandes. (Drauta, 2020)

1.5.Arquitectura Cliente-Servidor

Cliente-Servidor es uno de los estilos arquitectónicos distribuidos más conocidos, el cual está compuesto por dos componentes, el proveedor y el consumidor. El proveedor es un servidor que brinda una serie de servicios o recursos los cuales son consumido por el Cliente.

En una arquitectura Cliente-Servidor existe un servidor y múltiples clientes que se conectan al servidor para recuperar todos los recursos necesarios para funcionar, en este sentido, el cliente solo es una capa para representar los datos y se detonan acciones para modificar el estado del servidor, mientras que el servidor es el que hace todo el trabajo pesado.

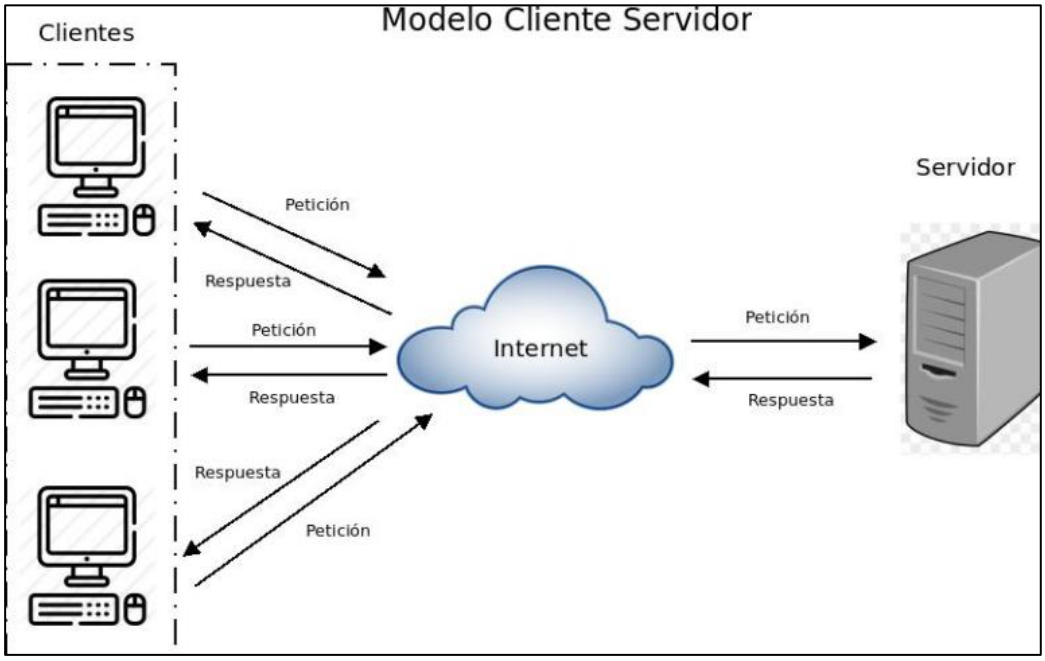
En esta arquitectura, el servidor deberá exponer un mecanismo que permite a los clientes conectarse, que por lo general es TCP/IP, esta comunicación permitirá una comunicación continua y bidireccional, de tal forma que el cliente puede enviar y recibir datos del servidor y viceversa.

Cliente-Servidor es considerada una arquitectura distribuida debido a que el servidor y el cliente se encuentran distribuidos en diferentes equipos (aunque podrían estar en la misma máquina) y se comunican únicamente por medio de la RED o Internet. (reactiveprogramming, 2020).

En la figura 1 se muestra de forma gráfica el funcionamiento de la arquitectura cliente servidor.

Figura 1

Arquitectura Cliente-Servidor



Nota: Representación de como el cliente realiza llamadas al servidor. Fuente: (notatecnologica, 2020)

1.6.Node JS

Node.js es un entorno de ejecución de un solo hilo, de código abierto y multiplataforma para crear aplicaciones de red y del lado del servidor rápidas y escalables. Se ejecuta en el motor de ejecución de JavaScript V8, y utiliza una arquitectura de E/S basada en eventos y sin bloqueos, lo que la hace eficiente y adecuada para aplicaciones en tiempo real.

Node.js utiliza la arquitectura «Single Threaded Event Loop» para manejar múltiples clientes al mismo tiempo. Para entender en qué se diferencia de otros tiempos de ejecución, tenemos que entender cómo se manejan los clientes concurrentes multihilo en lenguajes como Java.

En un modelo de solicitud-respuesta multihilo, varios clientes envían una solicitud y el servidor procesa cada una de ellas antes de devolver la respuesta. Sin embargo, se utilizan múltiples hilos para procesar las llamadas concurrentes. Estos hilos se definen en un pool de

hilos, y cada vez que llega una petición, se asigna un hilo individual para manejarla. (kinsta, 2021)

1.7.Node Package Manager

De sus siglas NPM (Node Package Manager) es un gestor de paquetes desarrollado en JavaScript, a través del cual podemos obtener cualquier librería con tan solo una sencilla línea de código, lo cual nos permitirá agregar dependencias de forma simple, distribuir paquetes y administrar eficazmente tanto los módulos como el proyecto a desarrollar en general.

Así mismo, es de gran importancia mencionar que Node.JS desde su versión 0.6.3 se instala automáticamente desde el entorno NPM, permitiendo a los desarrolladores instalar aplicaciones Node que se encuentren en el repositorio. En este mismo orden de ideas, cabe destacar que al instalar paquetes nuevos estos serán almacenados localmente en la carpeta que lleva por nombre "node_modules" dentro de nuestro proyecto, sin embargo, el desarrollador puede indicarle a NPM que instale dicho paquete de forma global, según lo considere necesario. (Yanina, 2019)

1.8.Visual studio code

Visual Studio Code es un editor de código fuente ligero pero potente que se ejecuta en su escritorio y está disponible para Windows, macOS y Linux. Viene con soporte integrado para JavaScript, TypeScript y Node.js y tiene un rico ecosistema de extensiones para otros lenguajes (como C ++, C #, Java, Python, PHP, Go) y tiempos de ejecución (como .NET y Unity). (Visual Studio Code, 2015)

1.9.Bootstrap

Bootstrap es un framework CSS desarrollado por Twitter en 2010, para estandarizar las herramientas de la compañía.

Inicialmente, se llamó Twitter Blueprint y, un poco más tarde, en 2011, se transformó en código abierto y su nombre cambió para Bootstrap. Desde entonces fue actualizado varias veces y ya se encuentra en la versión 4.4.

El framework combina CSS y JavaScript para estilizar los elementos de una página HTML. Permite mucho más que, simplemente, cambiar el color de los botones y los enlaces.

Esta es una herramienta que proporciona interactividad en la página, por lo que ofrece una serie de componentes que facilitan la comunicación con el usuario, como menús de navegación, controles de página, barras de progreso y más.

Además de todas las características que ofrece el framework, su principal objetivo es permitir la construcción de sitios web responsive para dispositivos móviles.

Esto significa que las páginas están diseñadas para funcionar en desktop, tablets y smartphones, de una manera muy simple y organizada. (rockcontent, 2020)

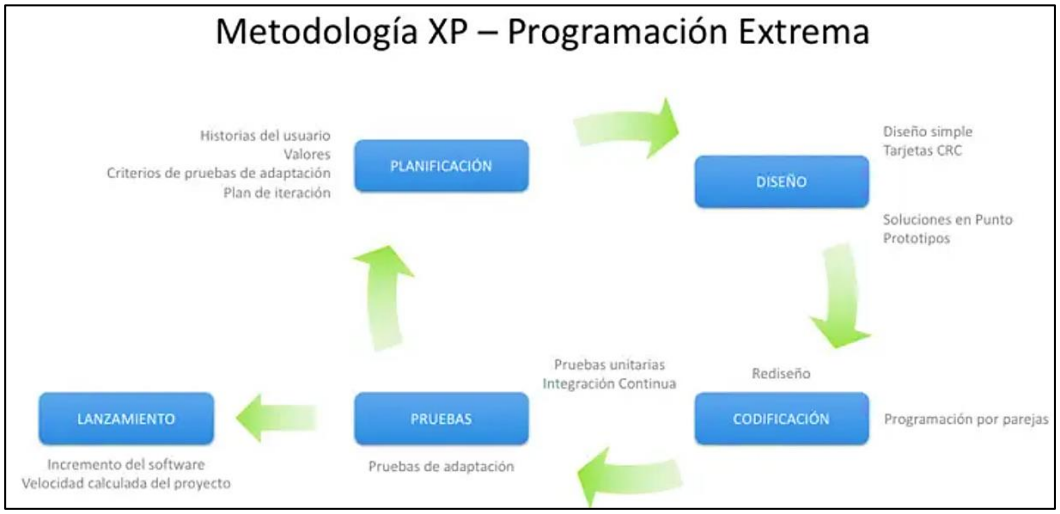
1.10. Metodología XP

La metodología XP es una metodología ágil para el desarrollo de software, brinda agilidad y flexibilidad en la gestión de proyectos. Se enfoca en desarrollar un producto según los requisitos del cliente. La flexibilidad de esta metodología permite realizar cambios constantes en el proyecto, esto brinda una calidad óptima del producto, ya que, con los cambios, el cliente va a tener mejores resultados en el desarrollo del producto, estos cambios se pueden realizar a lo largo del ciclo de vida de software. (sinnaps, 2020)

En la figura 1 se muestra de forma gráfica el funcionamiento de la metodología XP la misma que es utilizada en este proyecto.

Figura 2

Metodología XP



Nota: Representación del flujo que realiza la metodologia XP para desarrollar proyectos. Fuente: (Calvo, 2018)

CAPÍTULO 2

ANÁLISIS Y DISEÑO

2.1.ETAPA DE ANÁLISIS

2.1.1. Requerimientos del usuario

Los requerimientos permiten recopilar, analizar y verificar las necesidades de forma detallada que debe tener el sistema, en este caso la plataforma web, esto se consigue a través de los requisitos del sistema permitiendo cubrir las necesidades del cliente.

Disponer de una plataforma web para que los pequeños agricultores registren y comercialicen sus productos agrícolas de forma directa con los consumidores finales.

En forma complementaria facilitar a los pequeños emprendedores de una aplicación móvil que les permitan comercializar sus productos en forma complementaria a la plataforma web.

A través del uso de la tecnología proporcionar a los emprendedores de herramientas que les permita comercializar en línea sus productos en forma directa con los consumidores a fin de que reciban un precio justo por su producto.

2.1.2. Requerimientos funcionales y no funcionales

2.1.2.1.Requisitos funcionales

Los requisitos funcionales para este sistema están dados en función de los requerimientos del usuario y se tomó en cuenta el rol o función que llevará a cabo los diferentes tipos de usuarios que tendrá el sistema: invitado, usuario registrado y administrador.

Tabla 1

Requisitos funcionales

DESCRIPCIÓN
Habitar un módulo para el registro de nuevos usuarios.
Todos los usuarios registrados pueden almacenar productos y realizar pedidos.
Cada uno de los usuarios registrados puedan modificar sus datos.
Desde el sistema se gestiona cada uno de los productos.
El sistema debe permitir a los usuarios buscar y consultar sobre los productos agregados en el sistema.
Se permitirá el registro de productos al carrito siempre y cuando el usuario inicie su sesión.
El sistema debe almacenar la información de las ventas realizadas, para tener un registro de los productos que fueron vendidos.
El sistema debe permitir descargar la orden de compra al usuario con la información de cada uno de los productos que adquirió al realiza el pedido.
La orden de compra debe de estar en formato pdf.
El sistema enviará un correo electrónico cuando se registre alguna de las siguientes acciones: registro de nuevo usuario, recuperación de contraseñas.
El sistema debe permitir el registro de nuevos campos o atributos para categorizar productos.

Nota: Tabla de requerimientos funcionales que tendrá la plataforma web. Elaborado por: Los autores

2.1.2.2. Requerimientos no funcionales

Son aquellos que no se refieren a lo que hace el sistema si no como lo hacen, es decir detallan los criterios para evaluar las cualidades y restricciones del sistema, para este caso se nombra las siguientes, cómo se puede observar en la tabla 2.

Tabla 2

Requisitos no funcionales

DESCRIPCIÓN
El sistema delimita los privilegios y vistas con las que puede interactuar el administrador y el usuario registrado.
Realizar la validación de los datos ingresados en el sistema.
Interfaz gráfica de fácil lectura y amigable.

Nota: Tabla de requisitos no funcionales que tendrá la plataforma web. Elaborado por: Los autores

2.2.DISEÑO DEL SISTEMA

2.2.1. Descripción de actores

Se ha identificado tres actores que desempeñan tareas diferentes en el sistema, estos serán descritos en la tabla 3.

Tabla 3

Nombres de actores y su rol o perfil

PERFIL	DESCRIPCIÓN
Invitado	Persona que puede ver los productos agregados al sistema y ver la información sobre la plataforma web. Tiene acceso restringido a la realización de los pedidos y tampoco podrá realizar ventas.
Usuario Registrado	Persona que se encuentra registrada y tiene una cuenta en el sistema, puede realizar compras y publicar productos. Además, puede actualizar su información personal, ver su inventario, ver sus ventas y consultar sus compras realizadas.
Administrador	Persona encargada de administrar el sistema y el mantenimiento de información. Además, puede gestionar a los usuarios registrados y también

	puede realizar actividades como actualizar su información personal y la compra y venta de productos.

Nota: Los perfiles mencionados son los que van a interactuar con la plataforma web. Elaborado por: Los autores

Nombre de actores y los servicios que solicitan al sistema

En la Tabla 4 se puede observar los actores del sistema con sus respectivas tareas.

Tabla 4

Roles que intervienen en la aplicación web

ACTOR	NOMBRE DEL SERVICIO
Invitado	Acceder a información acerca de la plataforma web. Consulta productos.
Usuario Registrado	Consultar productos. Consultar sus compras y ventas. Gestionar productos. Gestionar pedidos. Actualizar información de su cuenta.
Administrador	Gestionar usuarios registrados. Recuperar cuentas caducadas. Gestionar usuarios administradores. Gestiona el mantenimiento de la plataforma. Realizar las mismas actividades que un usuario registrado.

Nota: Cada usuario podrá interactuar con los diferentes módulos dependiendo el perfil que este posea dentro del sistema. Elaborado por: Los autores

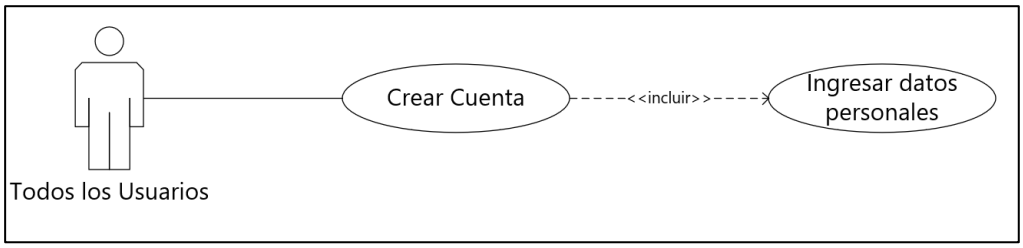
2.2.2. Diagramas de caso de uso

Casos de uso: Creación de una cuenta de usuario

En la figura 3, se puede visualizar el comportamiento (servicio) que el sistema informático propuesto tendrá frente al usuario (actor) al momento de solicitar al sistema la creación de una cuenta, donde las actividades a llevar a cabo se describen en la Tabla 5.

Figura 3

Casos de uso asociados a la creación de una cuenta



Nota: El diagrama de caso de uso muestra la creación de usuarios, esto se aplica a todos los usuarios que sean nuevos en el sistema. Elaborado por: Los autores

Escenario de casos de uso: Creación cuenta

Tabla 5

Creación cuenta

Nombre	Creación de cuenta
Descripción	Permitir la creación de una cuenta a nuevos usuarios
Actor	Todos los usuarios
Flujo Normal	Se muestra un formulario que el usuario debe completarlo. El usuario llena cada uno de los campos del formulario. El sistema guarda satisfactoriamente los datos. El usuario puede ingresar al sistema
Flujo secundario	Valida los datos y se muestra una notificación si los datos no son correctamente ingresados. El sistema detecta si existen campos vacíos e informa al usuario cuales son los campos que debe completar.

Interfaz	Registro de usuarios

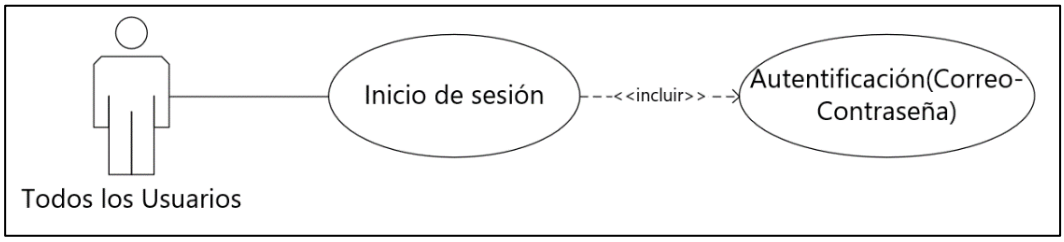
Nota: Se detalla la creación de la cuenta. Elaborado por: Los autores

Casos de uso: Inicio de sesión

En la figura 4, se puede visualizar el comportamiento (servicio) que el sistema informático propuesto tendrá frente al usuario (actor) al momento de ingresar al sistema, donde las actividades a llevar a cabo se describen en la Tabla 6.

Figura 4

Casos de uso asociados al inicio de sesión



Nota: El diagrama muestra el inicio de sesión, de cada uno de los usuarios. Elaborado por: Los autores

Escenario de casos de uso: Inicio de sesión

Tabla 6

Inicio de sesión

Nombre	Inicio de sesión
Descripción	Permitir el ingreso al sistema introduciendo su correo y contraseña.
Actor	Todos los usuarios
Flujo Normal	El usuario completa los campos, donde se ingresa el correo con su respetiva contraseña. Los datos son verificados en la base de datos. El usuario accede y visualiza su perfil.

Flujo Secundario	La aplicación verifica si el correo y la contraseña sean correctos, de no ser los datos correctos, se muestra un mensaje de error. El sistema detecta si existen campos vacíos e informa al usuario cuales son los campos que debe completar.
Interfaz	Inicio de sesión

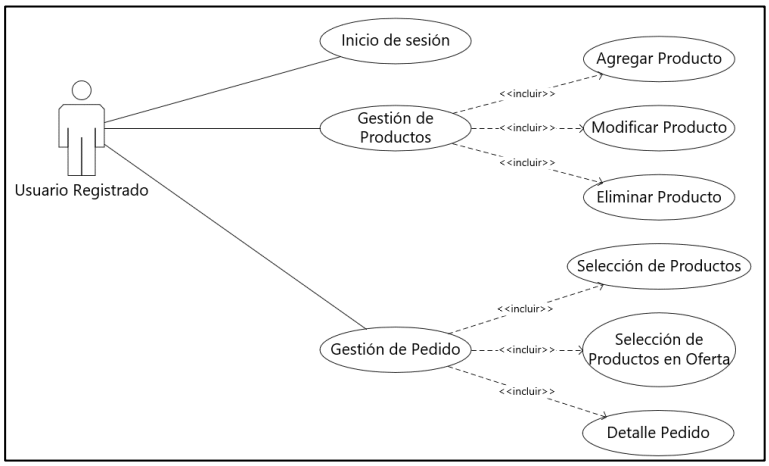
Nota: Se detalla el flujo que sigue el usuario para ingresar al sistema. Elaborado por: Los autores

Casos de uso: Usuario registrado

En la figura 5, se puede visualizar el comportamiento que tiene el usuario registrado en el sistema teniendo como rol la gestión de productos y la gestión de pedidos, esto serán descritos en la Tabla 7 y 8.

Figura 5

Casos de uso asociados a la gestión de pedidos y productos



Nota: El presente diagrama muestra los módulos con los cuales el usuario registrado puede interactuar. Elaborado por: Los autores

Escenario de casos de uso: Gestionar productos

Tabla 7

Gestión de productos

Nombre	Gestión de productos
Descripción	Permitir al usuario registrado ingresar los productos, modificar los productos registrados y eliminar de ser necesario.
Actor	Usuario Registrado y Administradores
Flujo Normal	Se muestra el formulario para agregar la información del producto a registrar. Una vez ingresados los datos se envía a grabar. Se informa del proceso realizado cuando se almacenan los datos. Si un usuario quiere modificar o actualizar la información selecciona el botón "Editar". Edita los campos que necesita modifica. Se informa al usuario del proceso realizado. Si desea eliminar el producto, escoge el producto y escoge la opción Eliminar. El sistema elimina satisfactoriamente el producto e informa al usuario del proceso realizado. Se actualiza los productos en el sistema. El usuario registrado ve las ventas realizadas.
Flujo secundario	Los datos son validados, si son incorrectos se despliega un mensaje de error. El sistema detecta si existen campos vacíos e informa al usuario cuales son los campos que debe completar.
Interfaz	Gestión de productos

Nota: La tabla detalla el flujo que sigue el usuario para gestionar los productos. Elaborado por: Los autores

Escenario de casos de uso: Gestionar pedidos

Tabla 8

Gestionar pedido

Nombre	Gestionar pedidos
Descripción	Permite al usuario registrado consultar, agregar, modificar y eliminar productos en el carrito.
Actor	Usuario Registrado y Administrador
Flujo Normal	En la barra de menú se dispone de un campo "Buscar", en la cual el usuario puede localizar los productos por su nombre. Se selecciona filtro de búsqueda por sector categoría o promoción. Seleccionar el producto de interés y añadirlo. El sistema almacenará los productos escogidos y los mostrará en el módulo "carrito". Si desea retirar el producto del carrito de compras seleccione el botón "Eliminar". Si desea agregar o disminuir la cantidad de producto, escoge el producto y lo incrementa o decrementa. Se actualiza los productos en el carrito. Se genera el pedido de los productos agregados al carrito. Descargar la orden de compra.
Flujo secundario	Si existe algún error al guardar la compra en el sistema, este no los guarda e informa del problema al usuario.
Interfaz	Carrito de compra

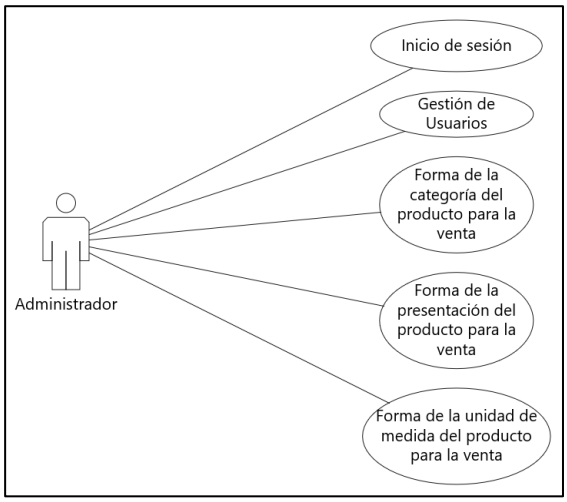
Nota: Se muestra el flujo que un usuario debe seguir para gestionar los pedidos. Elaborado por: Los autores

Casos de uso: Administrador

En la figura 6, se puede visualizar los diferentes casos de uso que el usuario administrador va a interactuar, dispone de permisos especiales, encargado del funcionamiento del sistema y serán descritas en la Tabla 9.

Figura 6

Casos de uso para usuarios administradores



Nota: Se muestra los módulos con los cuales el administrador puede interactuar. Elaborado por: Los autores

Escenario de casos de uso: Administración

Tabla 9

Proceso de administración

Nombre	Proceso de administración
Descripción	Permitir al administrador consultar y activar usuarios inactivos, también el agregar y eliminar categorías, unidades de medida, presentaciones para los productos y brinda permisos de administrador.
Actor	Administrador
Flujo Normal	El administrador accede al módulo administrador El administrador tiene alternativas de modifica el estado del usuario o agregar y eliminar atributos como son categoría, medidas, presentación, información.

	Si de desea activar la cuenta de un usuario que desactivo por inactividad o decisión del administrador, escoge la opción usuarios. La opción usuarios tiene un filtro de búsqueda para localizar todos los usuarios desactivados o un usuario en específico. Al localizar al usuario selecciona la opción activar El sistema mostrará un mensaje indicando que el usuario fue activado. Si el administrador desea desactivar un usuario, selecciona el botón desactivar usuario. Se debe ingresar el nombre del usuario el buscador para localizar Al localizar al usuario seleccione la opción desactivar. El sistema mostrara un mensaje indicando que el usuario fue desactivado. Si el administrador desea agregar un atributo para los productos del sistema selecciona una de las opciones como son categoría, unidad de medida y presentación. Se agrega el nombre del nuevo atributo y una descripción en el caso de categoría. Se agrega el nuevo atributo y el sistema los muestra en la tabla. Si el usuario desea agregar información con respecto a los administradores de la página. El sistema muestra información de los colaboradores de la plataforma y la opción de agregar uno nuevo. Se llena el formulario y se agrega El sistema actualiza los colaboradores de la plataforma.
Flujo secundario	En caso de dejar un campo vacío se mostrará un mensaje de llenar ese campo

Interfaz	Administración

Nota: Se detalla como el administrador gestiona la plataforma web. Elaborado por: Los autores

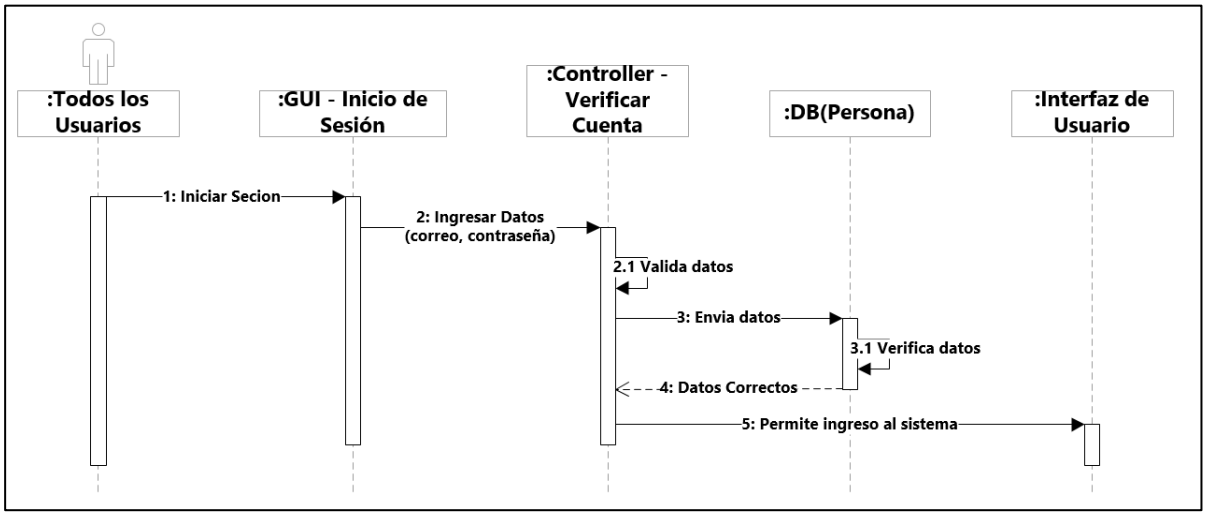
2.2.3. Diagrama de secuencia

Diagrama de secuencia: Inicio de sesión

Para una mejor apreciación de cómo funciona el inicio de sesión, el siguiente diagrama de secuencia se muestra cómo cada componente del sistema responde a las acciones del usuario como se representa en la figura 7.

Figura 7

Diagrama de secuencia de inicio de sesión



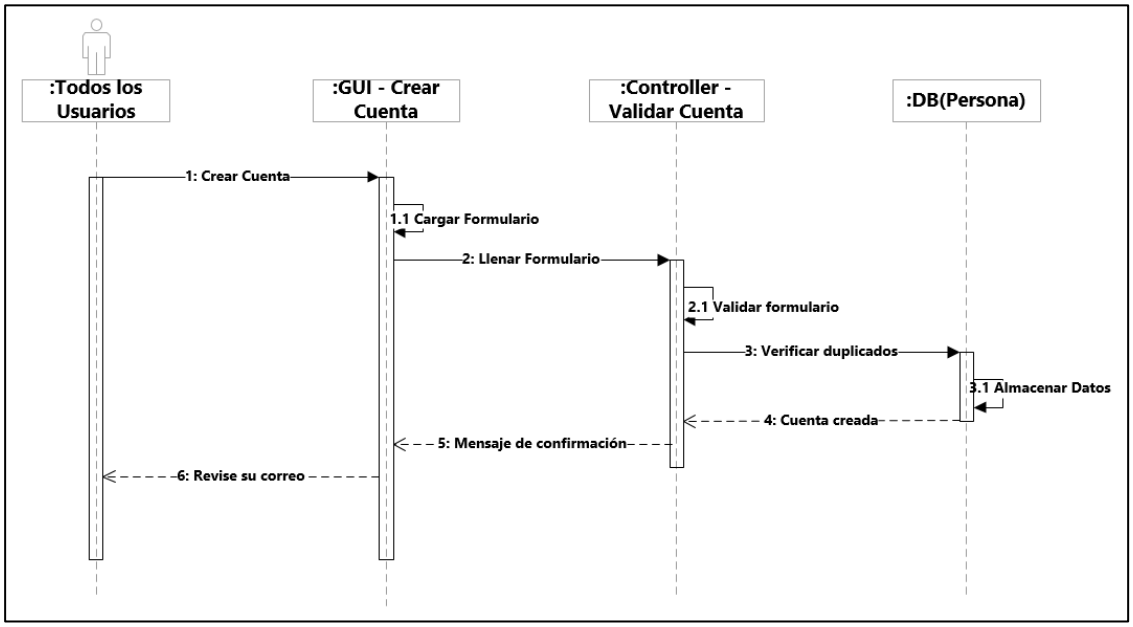
Nota: Diagrama de secuencia de inicio de sesión y como este interactúa con el usuario. Elaborado por: Los autores

Diagrama de secuencia: Crear cuenta de usuario

En la figura 8 se representa cómo funciona la creación de usuario, correspondiente al diagrama de secuencia de inicio de sesión y cómo cada componente del sistema responde a las acciones del usuario.

Figura 8

Despliegue del diagrama de secuencia crear cuenta



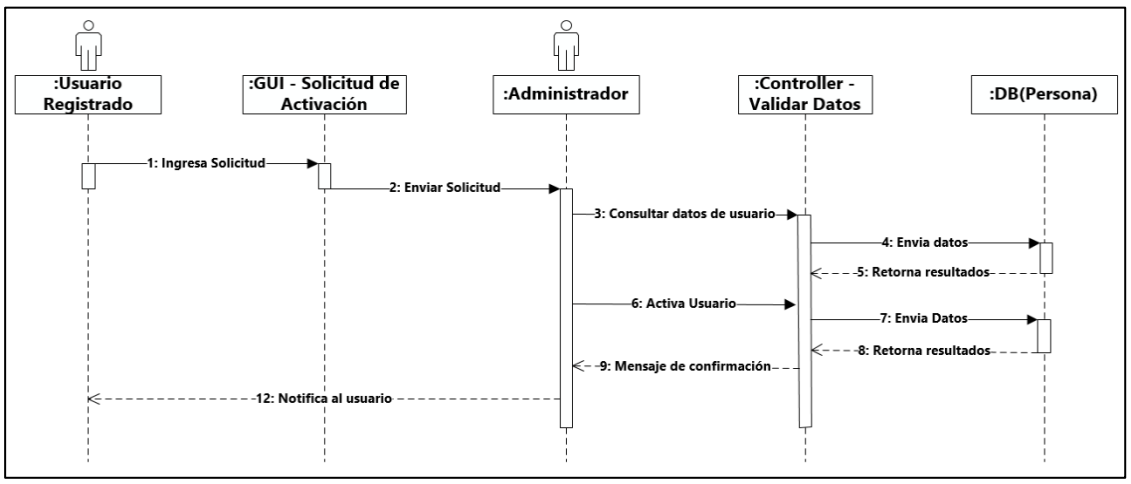
Nota: Diagrama de secuencia relacionado a la creación de un nuevo usuario. Elaborado por: Los autores

Diagrama de secuencia: Administrador

En la figura 9, el diagrama de secuencia muestra como el Administrador realiza el proceso de activación de una cuenta que se encuentra desactivada por inactividad.

Figura 9

Diagrama de secuencia activación de cuenta caducada

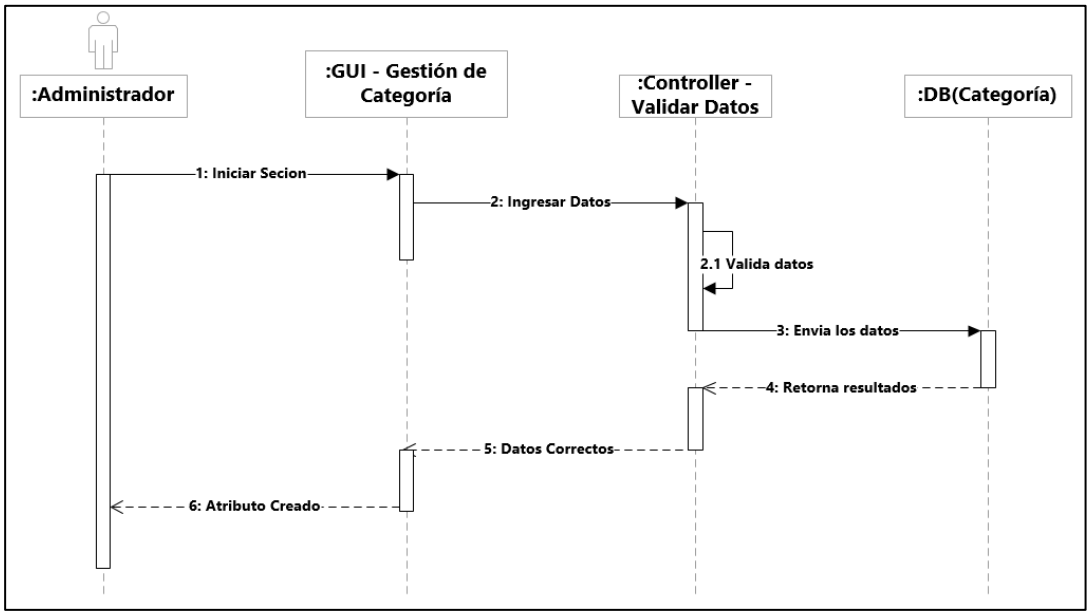


Nota: Diagrama de secuencia relacionado a la activación de cuenta caducada de un usuario. Elaborado por: Los autores

En la figura 10, el diagrama de secuencia gestión de categoría, detalla como el Administrador crea más atributos para categorizar.

Figura 10

Diagrama de secuencia gestión de categoría



Nota: Diagrama de secuencia para la gestión de categoría. Elaborado por: Los autores

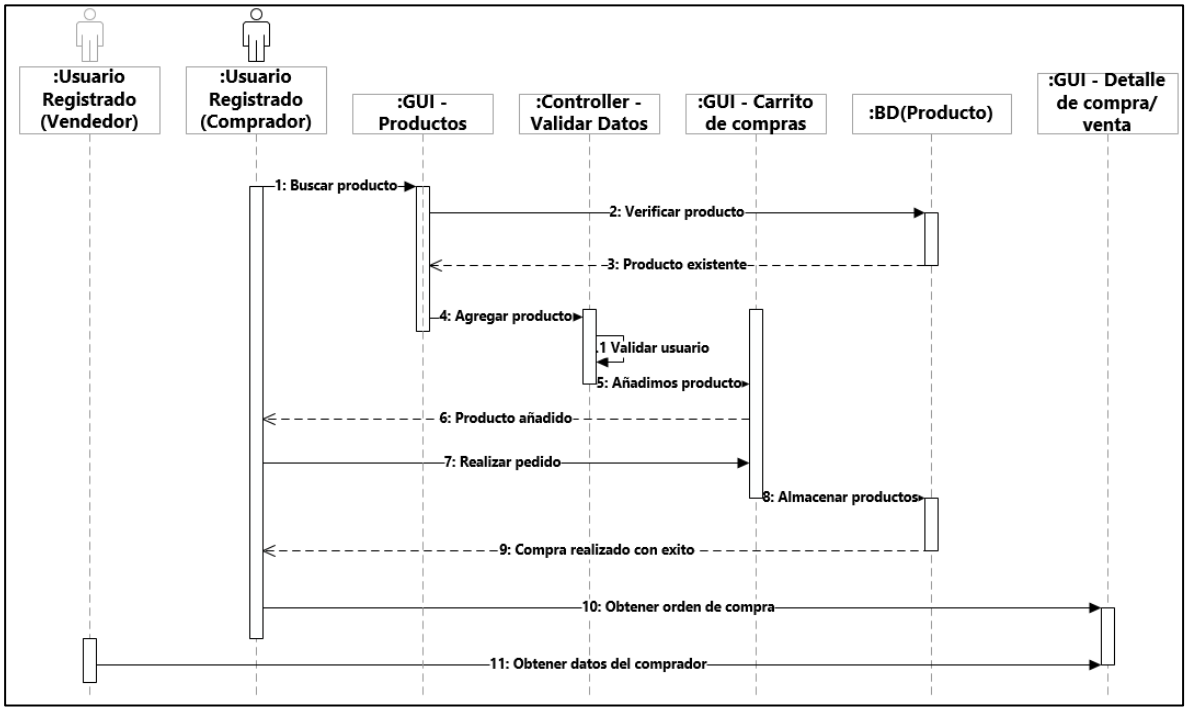
El mismo procedimiento mostrado en la figura 10 se aplica para la gestión de presentación y la gestión de unidad de medida.

Diagrama de secuencia: Gestionar pedidos

En la figura 11 se representa la gestión de pedidos, en donde se detalla como el usuario registrado interactúa con el sistema al realizar un pedido o compra.

Figura 11

Diagrama de secuencia Gestión de pedidos



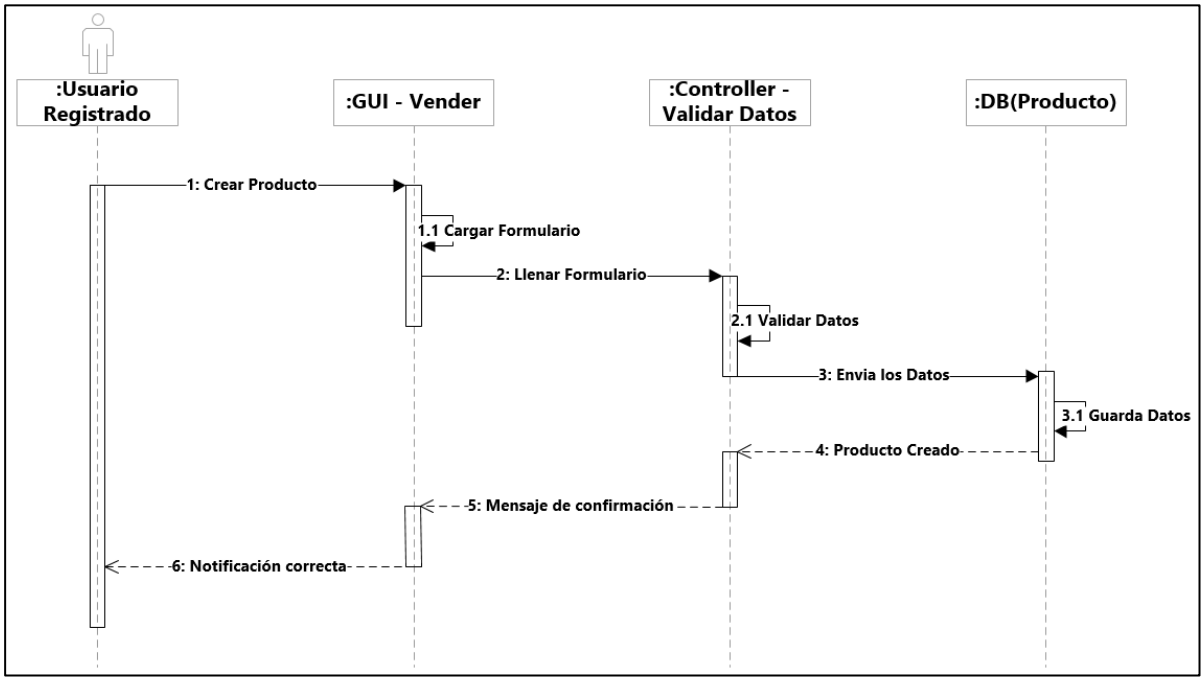
Nota: Diagrama de secuencia relacionado a la realización de un pedido. Elaborado por: Los autores

Diagrama de secuencia: Gestionar productos

A continuación, en la figura 12, detalla la secuencia que sigue el usuario al ingresar nuevos productos al sistema por parte del usuario registrado.

Figura 12

Despliegue de diagrama de secuencia Gestionar Productos



Nota: Se detalla la secuencia relacionada al ingreso de productos. Elaborado por: Los autores

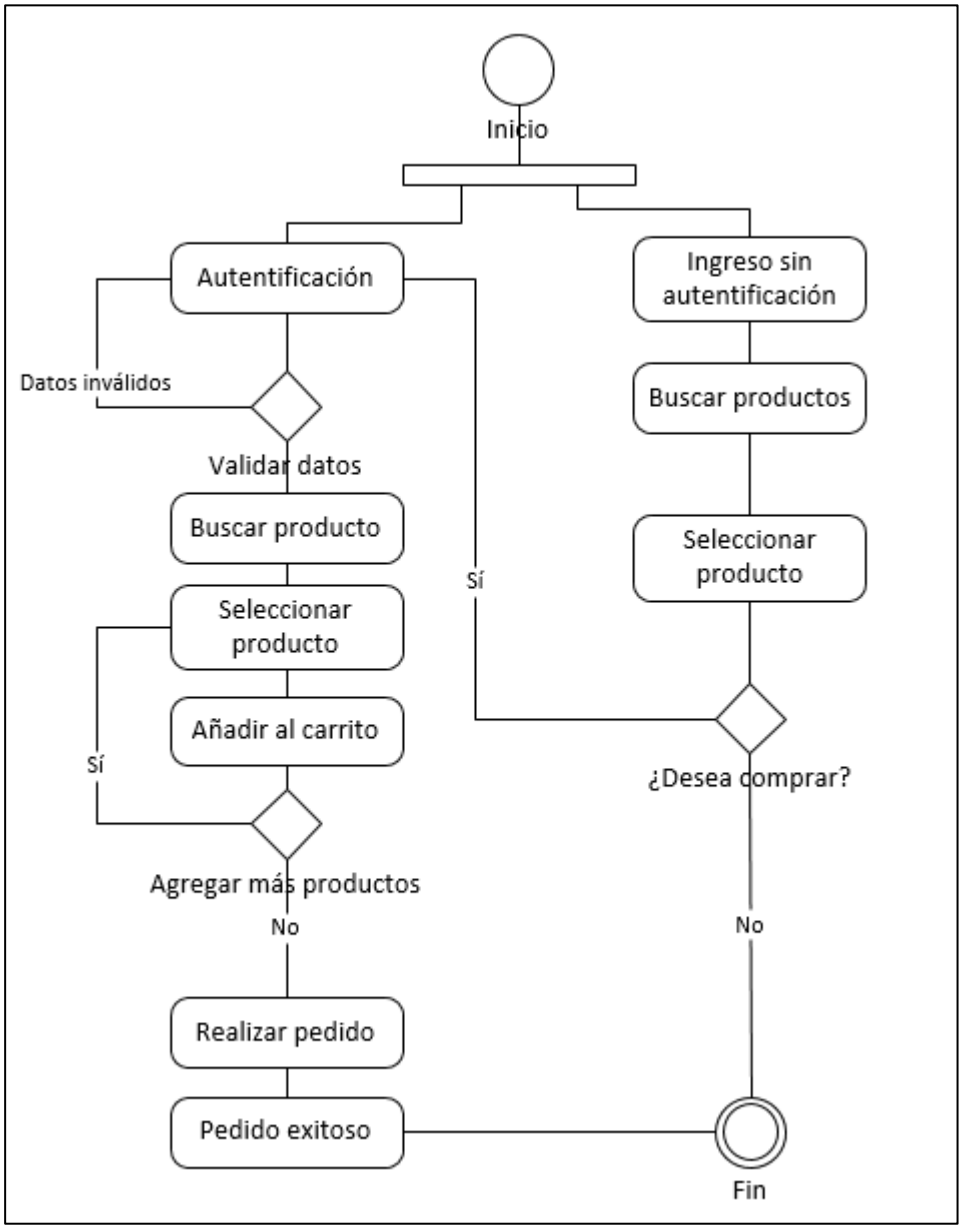
2.2.4. Diagrama de Actividades

Diagrama de actividades Gestionar Pedido

La figura 13 muestra el diagrama de actividades gestionar pedido, donde se detalla la secuencia que sigue el sistema al realizar un pedido o compra por parte del usuario registrado.

Figura 13

Diagrama de actividades Gestionar Pedido



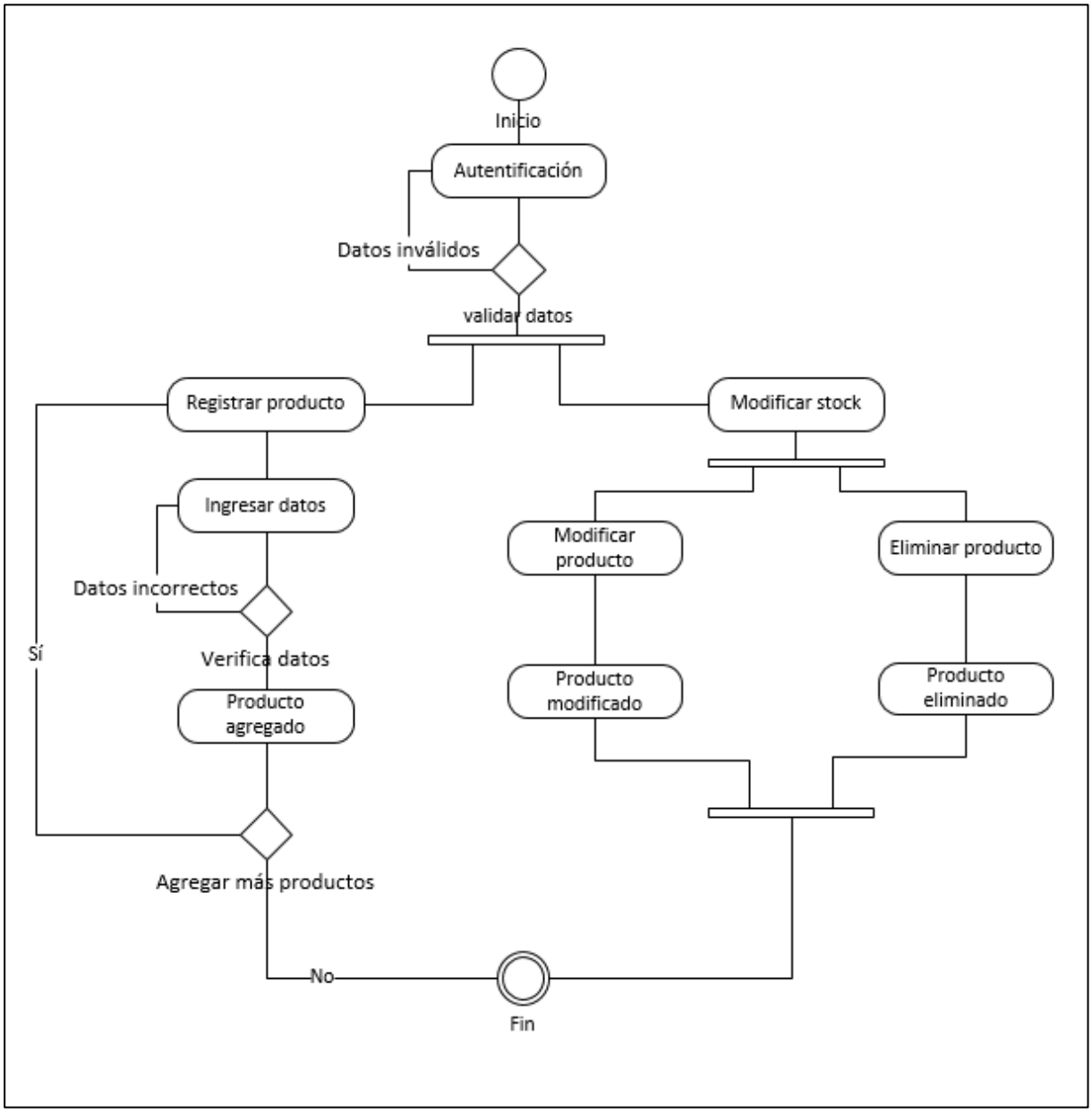
Nota: Diagrama donde se muestra la secuencia que realiza el usuario al realizar un pedido. Elaborado por: Los autores

Diagrama de actividades Gestionar Productos

La figura 14 muestra el diagrama de actividades de gestión de productos donde se detalla la secuencia que sigue el sistema al momento de gestionar los productos por parte del usuario registrado.

Figura 14

Diagrama de actividades gestionar productos



Nota: Diagrama donde se muestra la secuencia que realiza el usuario al gestionar sus productos.

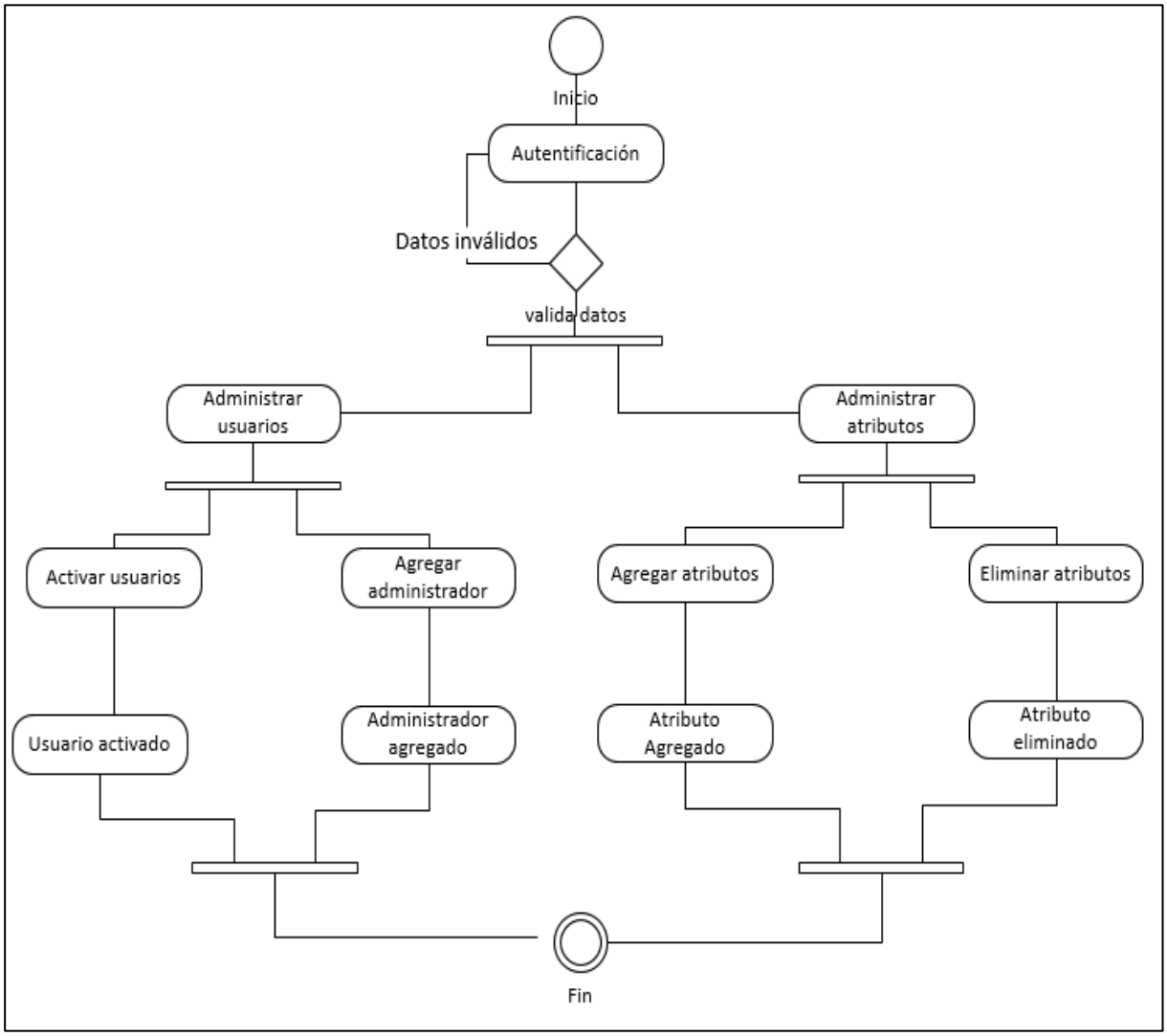
Elaborado por: Los autores

Diagrama de secuencia Administrador

La figura 15 muestra el diagrama de secuencia del Administrador, donde se detalla el proceso de activación de una cuenta o la creación de nuevos atributos para los productos.

Figura 15

Diagrama de secuencia Administrador



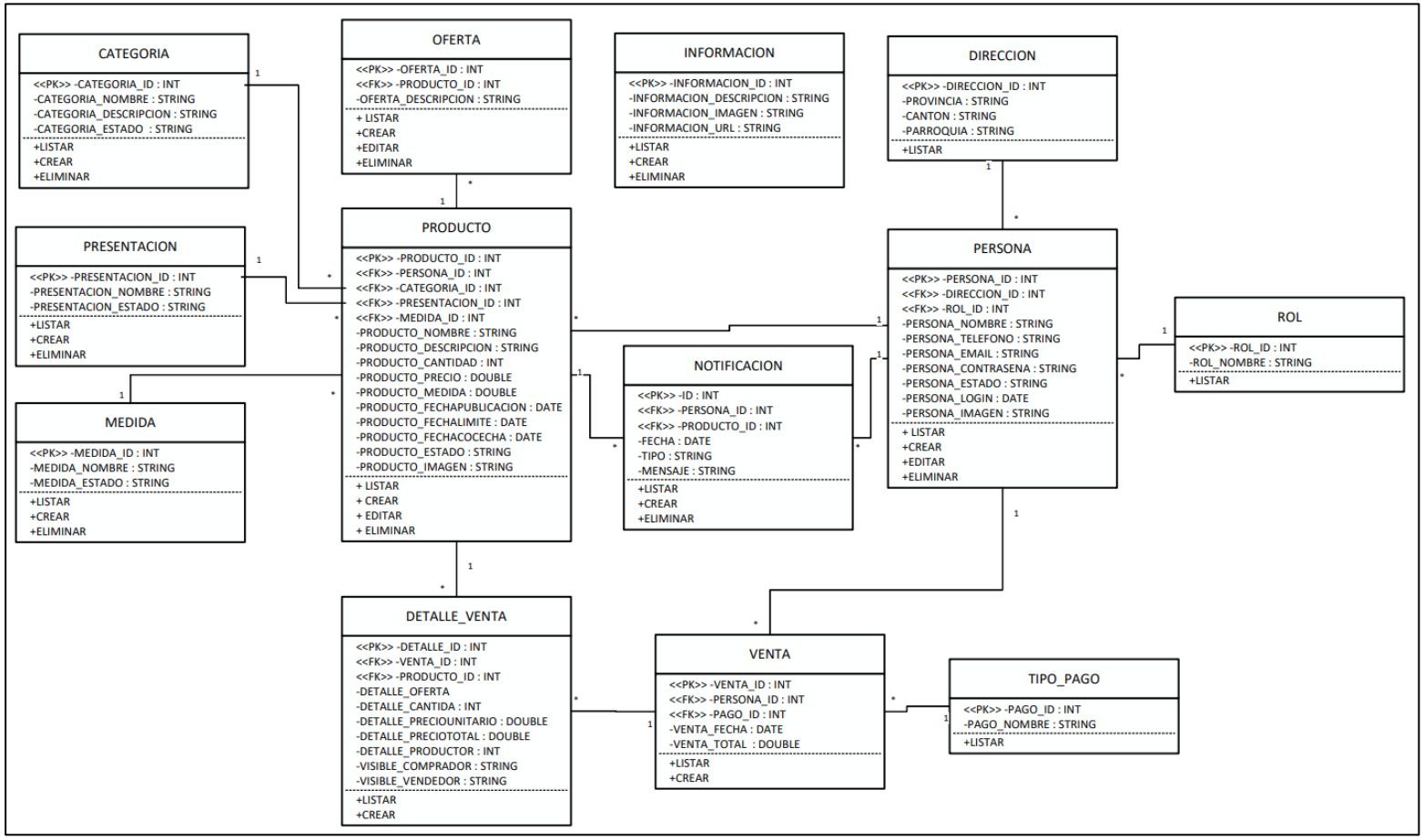
Nota: Diagrama donde se muestra la secuencia que realiza el administrador. Elaborado por: Los autores

2.2.5. Diagrama de clases de la plataforma web

Como se puede visualizar en la figura 16, se detalla las clases con sus atributos y la relaciones que existen en cada una de ellas.

Figura 16

Despliegue de clases

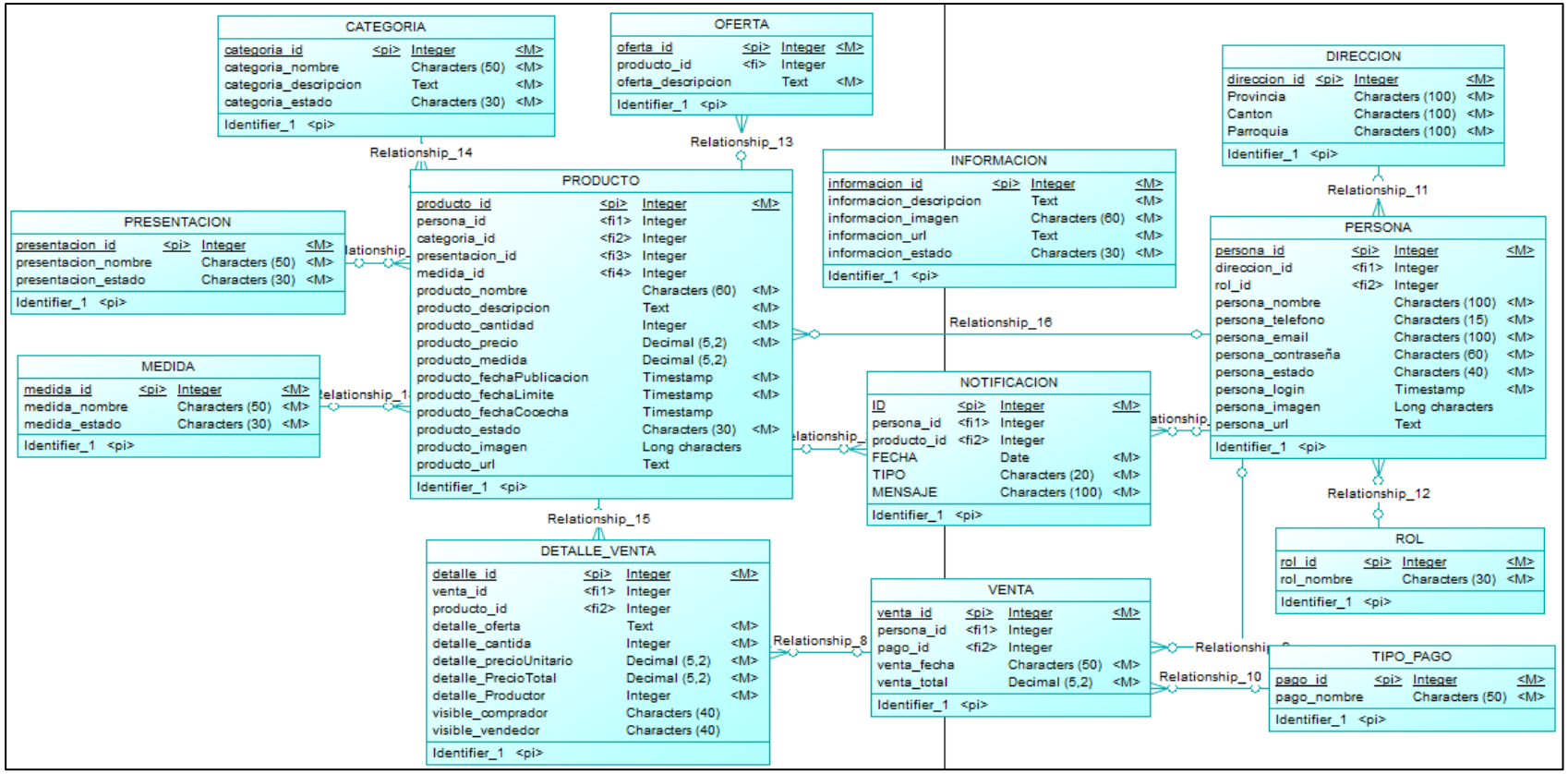


Nota: Diagrama donde se detalla los atributos y relaciones de cada clase. Elaborado por: Los autores

2.3.DISEÑO CONCEPTUAL DE BASE DE DATOS PARA LA APLICACIÓN WEB

Figura 17

Base de Datos perteneciente a la plataforma web



Nota: Base de datos donde se detalla las tablas y las relaciones que tiene cada una de ellas. Elaborado por: Los autores

2.4.PROTOTIPO DE LA INTERFAZ DE LA APLICACIÓN WEB

Para diseñar el prototipo y dar una propuesta se usó Balsamiq Wireframes, software que proporciona diferentes herramientas para crear prototipos de interfaz de forma eficaz e intuitiva.

Prototipo de la Pantalla principal

Para una mejor apreciación de cómo funciona la pantalla inicial, se diseñó un prototipo de pantalla de inicio donde se muestra diferentes componentes como la barra de menú compuesto por diferentes módulos como: Comprar, Inicio de sesión, Crear Cuenta e Información referente a la organización, esto es representado en la figura 18.

Figura 18

Prototipo Pantalla Inicio

Nota: Prototipo de la página de inicio y su barra de menú donde se encuentran diferentes módulos del sistema. Elaborado por: Los autores

Prototipo del Inicio de sesión

En la figura 19 se muestra el prototipo de la pantalla de inicio de sesión para que los usuarios registrados puedan acceder. En el formulario el usuario debe ingresar su correo y contraseña, estas credenciales serán verificadas en la base de datos, si son correctos el usuario podrá ingresar al sistema, caso contrario se desplegará el mensaje “Credenciales no válidas”.

Figura 19

Prototipo del ingreso al sistema

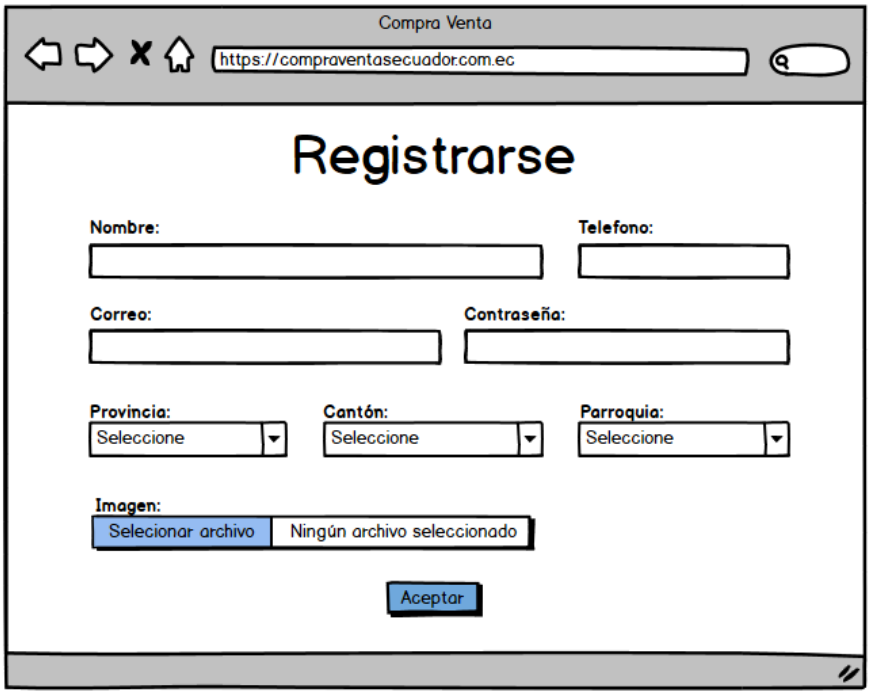
Nota: Prototipo donde se observa el ingreso al sistema o login. Elaborado por: Los autores

Prototipo del Formulario de registro

Prototipo donde se muestra el formulario de creación de la cuenta, donde cada usuario tendrá que registrar sus datos personales, una vez ingresado los datos, el sistema los verificará y en caso de no existir ningún error estos serán guardados en el sistema, para una mejor apreciación ver la figura 20.

Figura 20

Prototipo de creación de cuenta



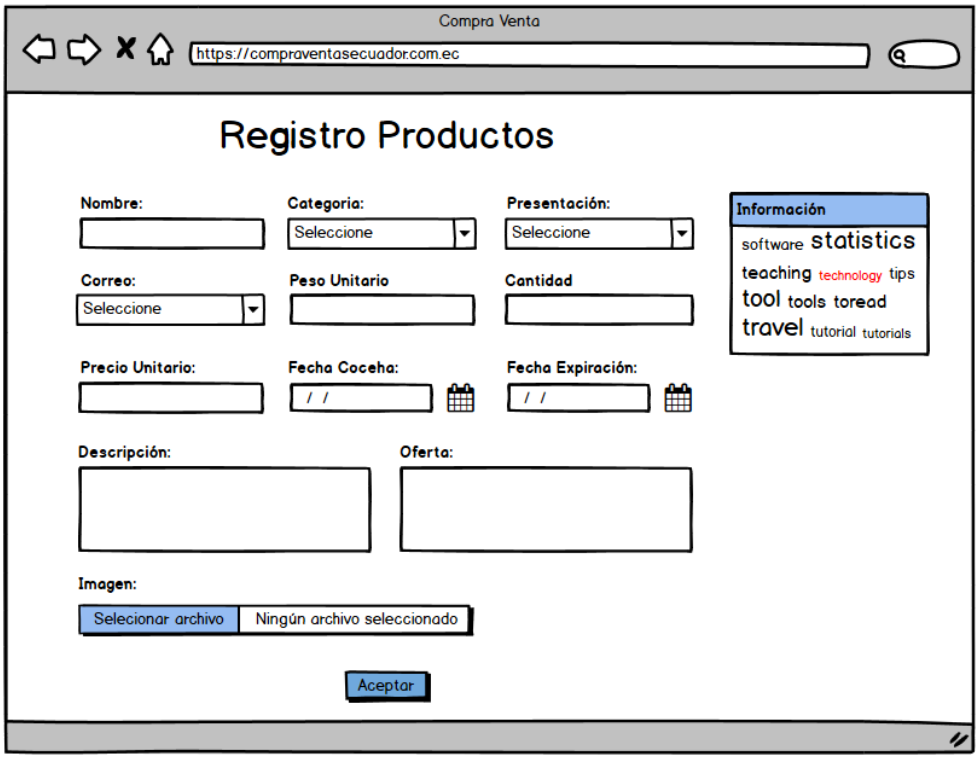
Nota: Prototipo del registro de creación de una cuenta. Elaborado por: Los autores

Prototipo Registro de productos

Prototipo de la pantalla del ingreso de productos, donde se muestra un formulario en el cual el usuario debe ingresar los datos solicitados referente al producto que se va a publicar, una vez ingresado los datos, el sistema los verificará, y en caso de no existir ningún error estos serán guardados en el sistema, para una mejor apreciación ver la figura 21.

Figura 21

Prototipo del ingreso de producto al sistema



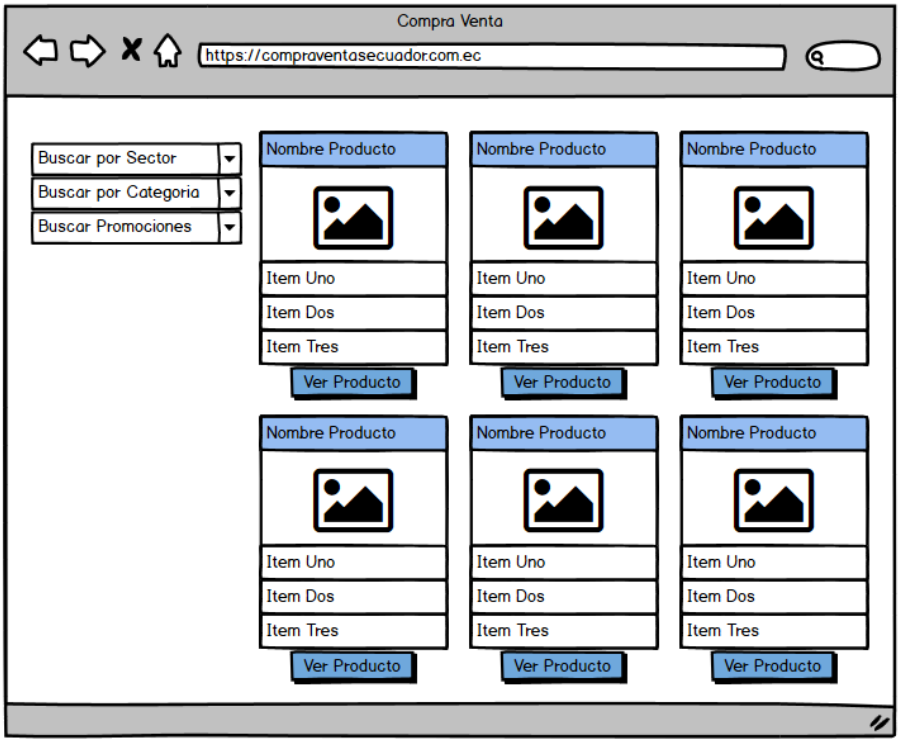
Nota: Prototipo del registro de nuevos productos al sistema. Elaborado por: Los autores

Prototipo Pantalla de Producto

Prototipo donde se presenta los productos que se venden en el sistema, los productos deben contener una imagen, nombre, precio, cantidad, presentación, peso, ofertas y la dirección, es la parte izquierda de la pantalla se encuentra 3 tipos de búsqueda por categorías y un buscador general de productos, todo esto se lo muestra en la figura 22.

Figura 22

Prototipo Pantalla de Productos



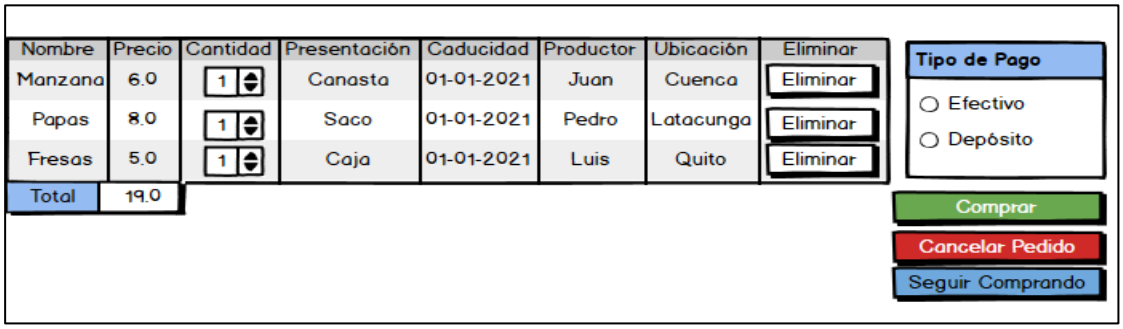
Nota: Bosquejo donde se presenta los productos que van a hacer ofertados en el sistema. Elaborado por: Los autores

Prototipo Pantalla Carrito

En la figura 23 se presenta el carrito de compras, en donde cada producto agregado al carrito debe permitir cambiar la cantidad o eliminar el producto del carrito, en esta pantalla debe mostrar toda la información relacionada con los productos que realizara el pedido y seleccionar el tipo de paga que el usuario va a realizar. Finalmente se describe el valor total generados por los productos agregados.

Figura 23

Prototipo Pantalla Carrito



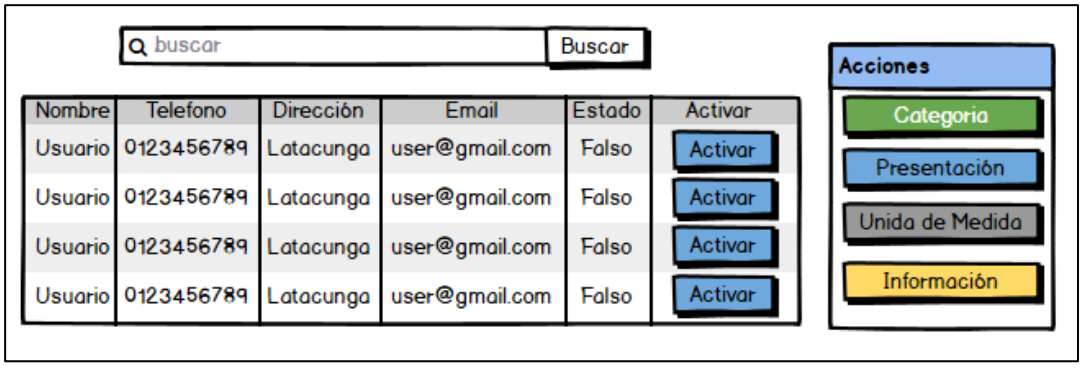
Nota: En la presente figura se muestra la ventana de carrito, solo se tendrá acceso a ella si se ha agregado un producto al mismo. Elaborado por: Los autores

Prototipo Pantalla Administrador

En el bosquejo que se muestra en la figura 24, se presenta la interfaz de la ventana administrador, en donde se hace referencia a diferentes módulos como: Categoría, Presentación, Unidad de medida e Información.

Figura 24

Prototipo Pantalla Administrador



Nota: En la presenta figura se muestra las diferentes actividades que va a desempeñar el administrador en el sistema. Elaborado por: Los autores

CAPÍTULO 3

CONSTRUCCIÓN

En el siguiente capítulo se muestra la construcción de la plataforma web, en donde se toma en cuenta a las personas que van a usar el servicio, las cuales no poseen muchas habilidades en computación, por lo que se pensó en una web que sea fácil de usar y que la interfaz sea lo más limpia para que no exista distracciones o ruido visual.

3.1. LIBRERÍAS Y FRAMEWORK USADOS EN EL SISTEMA

Librerías:

express-handlebars: Es un motor de visualización que permite renderizar las vistas.

express-session: Permite crear sesiones para cada usuario que son almacenadas en el servidor.

multer: Se utilizó para cargar archivos al sistema, en este caso imágenes.

fs: Administra y permite interactuar con el sistema de archivos.

mysql: Se utiliza para conectar la base de datos con la plataforma web.

passport: Permite autentificar solicitudes.

passport-local: Se utilizó para la autentificación sea de forma local.

pdfkit: Permite transformar consultas de base de datos a pdf.

yup: Permite validar los datos ingresados al sistema.

Md5: Algoritmo que permite la codificación de documentos o texto.

jsonwebtoken: Permite la creación de tokens de acceso.

Framework Express:

Se usa para la creación de aplicaciones web que corre a lado del servidor la cual está alojado dentro del entorno de ejecución NodeJS, está diseñado para crear aplicaciones en menos tiempo ya que nos proporciona funcionalidades como el enrutamiento, opciones para gestionar sesiones, facilidades para motores de plantillas (Pug, EJS, Handlebars), cookies, etc.

3.2. FUNCIONAMIENTO DE LA PÁGINA WEB

3.2.1. Pantalla de ingreso al sistema

Como se representa en la figura 25, el formulario de autentificación presentará el siguiente mensaje de error, si el correo o la contraseña ingresados son incorrectos.

Figura 25

Inicio de sesión

Nota: Pantalla de autenticación de la página web. Elaborado por: Los autores

En la figura 26 se muestra como la función “loginValidation” realiza la validación a los campos de correo y contraseña, verificando que el correo tenga la estructura de un correo y no se encuentre vacía y la contraseña contenga un mínimo de 5 caracteres y el campo no esté vacío.

Figura 26

Validación del login

```
43  function loginValidation(data) {
44      const schema = yup.object().shape({
45          email: yup.string()
46              .email('Correo electronico inválido')
47              .required('Campo email requerido'),
48
49          password: yup.string()
50              .min(5, 'Campo contraseña mínimo 5 caracteres')
51              .required('Campo contraseña requerido'),
52      });
53
54      schema.validateSync(data);
55  }
```

Nota: Bosquejo donde se muestra la función de la validación del login. Elaborado por: Los autores

3.2.2. Pantalla de registro de usuario

En el caso de que el usuario registre mal un campo el sistema le informará del error, por ejemplo, en el caso del campo teléfono solo podrá ingresar números, en el campo de correo electrónico si no es una dirección de correo electrónico el campo le indica que el dato ingresado no es un correo electrónico. Si el usuario deje en blanco un campo el sistema le informara de que todos los campos deben ser completados para realizar el registro en el sistema. Si uno del combo box donde se encuentre cargada la información de la dirección como son Provincia, Cuidad y Parroquia no se encuentra seleccionada, esta mostrará una etiqueta indicando donde se encuentra el error, como se visualiza en la figura 27.

Figura 27

Pantalla de creación de cuenta

Nota: Pantalla donde se muestra el formulario para la creación de una cuenta. Elaborado por: Los autores

La función “createUsersValidation” realiza toda la validación del formulario del ingreso de datos, validando que ingresen el tipo de variable correspondiente, el número mínimo requerido de caracteres debe ser de 8 para el campo teléfono y debe contener números, en el campo correo puede contener caracteres especiales como punto, letras y números, la contraseña debe tener un mínimo de 5 caracteres y finalmente el campo nombre debe tener un mínimo de 10 caracteres, esto se ve representado en la figura 28.

Figura 28

Validación de registro de usuario

```
function createUsersValidation(data) {
    const schema = yup.object().shape({
        PERSONA_NOMBRE: yup.string()
            .min(10, 'Campo nombre mínimo 10 caracteres')
            .matches(/^[a-zA-ZÀ-ÿ\u00f1\u00d1]+(\s*[a-zA-ZÀ-ÿ\u00f1\u00d1]*)*[a-zA-ZÀ-ÿ\u00f1\u00d1]+$/g,
            .required('Campo nombre requerido'),

        PERSONA_TELEFONO: yup.string()
            .min(8, 'Campo telefono mínimo 8 caracteres')
            .matches(/^[0-9]+$/, 'Campo telefono solo debe de tener números')
            .required('Campo telefono requerido'),

        PERSONA_EMAIL: yup.string()
            .email('Correo electronico inválido')
            .required('Campo email requerido'),

        PERSONA_CONTRASENA: yup.string()
            .min(5, 'Campo contraseña mínimo 5 caracteres')
            .required('Campo contraseña requerido'),

        DIRECCION_ID: yup.number()
            .moreThan(0, 'Error en el campo dirección')
            .required('Campo dirección requerido'),

    });

    schema.validateSync(data);
}
```

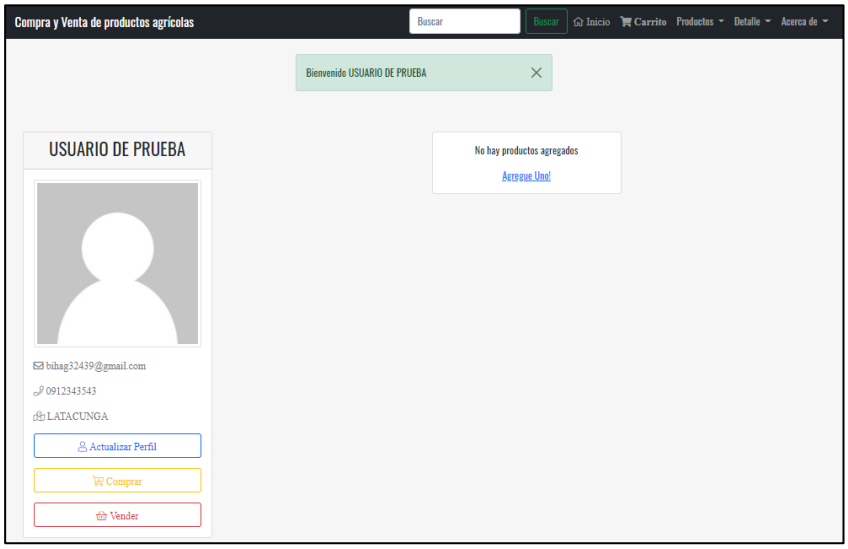
Nota: En la presente figura se puede observar la validación de la creación de la cuenta de usuario. Elaborado por: Los autores

3.2.3. Pantalla de usuario

Una vez que el usuario haya ingresado a su cuenta se desplegará la pantalla de usuario, la cual se presenta en la Figura 29, si el usuario no posee productos agregados se desplegara un mensaje de “No hay productos agregados” con un enlace que le redireccionara a la página de registro de productos. En la parte izquierda de la pantalla se muestra el perfil e información personal del usuario como son correo, teléfono y dirección, seguido se muestra tres botones, uno de estos es el botón de actualizar perfil que permitirá al usuario editar los datos personales. Los botones restantes son de comprar y venta de productos los cuales va a redirigir a las páginas correspondientes.

Figura 29

Pantalla de usuario



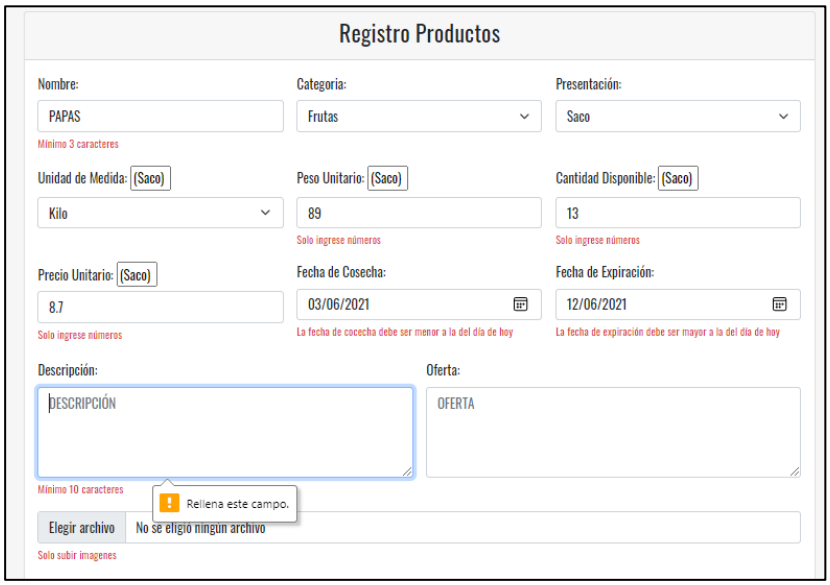
Nota: Pantalla donde se muestra la interfaz de usuario. Elaborado por: Los autores

3.2.4. Pantalla de registro de productos

En la Figura 30, se presenta la pantalla de registro de productos, en el caso de que el usuario registre mal un campo o lo deje en blanco el sistema le informara del error.

Figura 30

Pantalla de registro de productos



Nota: Ventana donde se muestra el formulario para el ingreso de productos al sistema. Elaborado por: Los autores

La función “createProductsValidation” realiza toda la validación del formulario del ingreso de datos de los productos, validando que ingresen el tipo de variable correspondiente, el número mínimo requerido de caracteres y que los campos no se encuentren en blanco, esto se ve representado en la figura 31.

Figura 31

Validación de registro de producto

```
function createProductsValidation(data) {
    const schema = yup.object().shape({
        PRODUCTO_NOMBRE: yup.string()
            .min(3, 'Campo nombre mínimo 3 caracteres')
            .matches(/^[a-zA-ZÀ-ÿ\u00f1\u00d1]+(\s*[a-zA-ZÀ-ÿ\u00f1\u00d1]*)*[a-zA-ZÀ-ÿ\u00f1\u00d1]+$/g, 'Campo nombre solo debe de tener letras')
            .required('Campo nombre requerido'),

        PRODUCTO_CANTIDAD: yup.string()
            .min(1, 'Campo cantidad mínimo 1 valor')
            .matches(/^[0-9]+$/, 'Campo cantidad solo debe de tener números')
            .required('Campo cantidad requerido'),

        PRODUCTO_PRECIO: yup.string()
            .min(1, 'Campo precio mínimo 1 valor')
            .matches(/^\d+(?:.\d+)?$/, 'Campo precio solo debe de tener números')
            .required('Campo precio requerido'),

        PRODUCTO_MEDIDA: yup.string()
            .min(1, 'Campo medida mínimo 1 valor')
            .matches(/^\d+(?:.\d+)?$/, 'Campo peso solo debe de tener números')
            .required('Campo peso requerido'),

        PRODUCTO_DESCRIPCION: yup.string()
            .min(10, 'Campo descripción debe tener mínimo 10 caracteres')
            .required('Campo contraseña requerido'),

        PRODUCTO_FECHACOCECHA: yup.date()
            .max(new Date(), 'El campo fecha de cocecha debe ser menor al del día de hoy')
            .required('Campo fecha de cocecha requerido'),

        PRODUCTO_FECHALIMITE: yup.date()
            .min(new Date(), 'El campo fecha limite debe ser mayor al del día de hoy')
            .required('Campo fecha limite requerido'),

        PRODUCTO_CANTIDAD: yup.number()
            .moreThan(0, 'Campo cantidad debe ser mayor a 0')
            .required('Campo dirección requerido'),
```

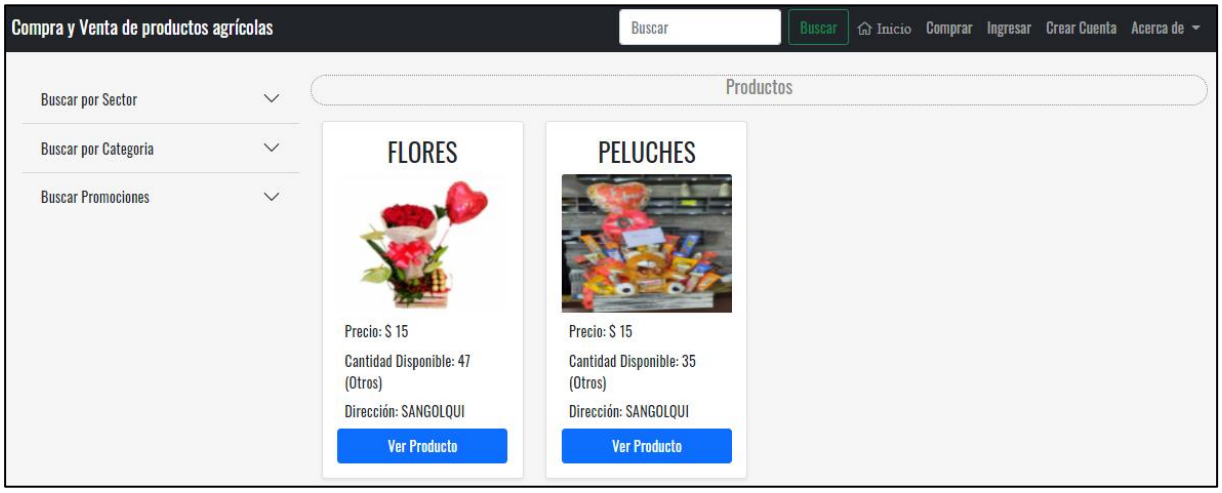
Nota: En la presente figura se puede observar la validación de la creación de un producto. Elaborado por: Los autores

3.2.5. Pantalla de productos

En la Figura 32, se puede observar los productos ofertados y disponibles en la plataforma web, si el usuario no accede a su cuenta, el usuario no puede agregar productos al carrito de compras y realizar su pedido.

Figura 32

Pantalla de productos



Nota: Pantalla de productos donde se visualiza todos los productos publicados. Elaborado por: Los autores

Trigger que permite la validación del ingreso de productos al carrito, validando que cada producto ingresado al carrito sea solo disponible si el usuario esta registrado, como se visualiza en la figura 33.

Figura 33

Validación de ingreso de productos al carrito

```
225         try {
226             if (req.user.PERSONA_ID) {
227                 const idUser = req.user.PERSONA_ID;
228                 Cart.save(productsId, addedProduct, idUser);
229                 res.redirect('/cart');
230             }
231         } catch {
232             req.flash('message', 'Primero Debe Iniciar Sesión Para Comprar');
233             res.redirect('/signin');
234         }
```

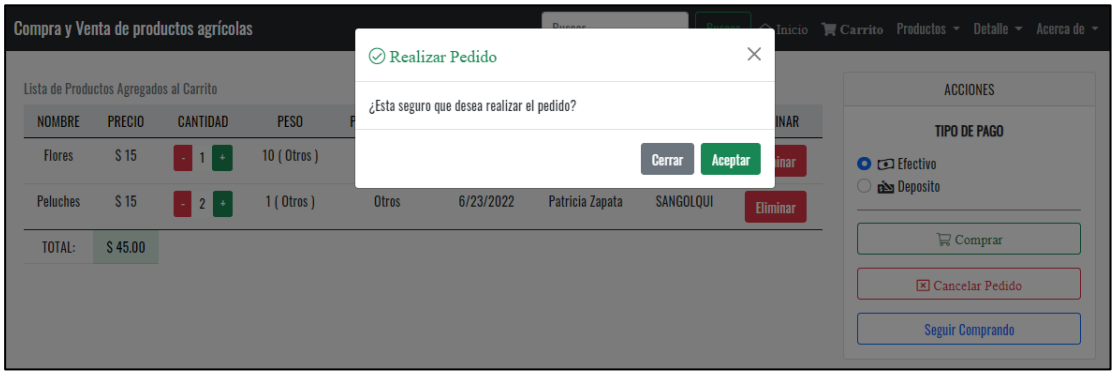
Nota: En la presente figura se puede observar la validación de la creación de un producto. Elaborado por: Los autores

3.2.6. Pantalla carrito

En la Figura 34, presenta los productos agregados al carrito, donde el usuario antes de realizar la compra o cancelar la operación se le presentará un mensaje de confirmación.

Figura 34

Pantalla carrito



Nota: Ventana donde se muestra los productos agregados al carrito de compras. Elaborado por: Los autores

En la Figura 35, se observa las variables en las que se van a guardar todos los datos enviados por el comprador.

Figura 35

Productos agregados al carrito

```
cart.push({
    idUser: userBuy,
    PRODUCTO_ID: productId,
    CATEGORIA_NOMBRE: productCategoria,
    PRESENTACION_NOMBRE: productPresentacion,
    MEDIDA_NOMBRE: productMedida,
    PRODUCTO_NOMBRE: productNombre,
    PRODUCTO_DESCRIPCION: productDescripcion,
    PRODUCTO_MEDIDA: productMedidaValor,
```

Nota: En la siguiente figura se puede observar las diferentes variables que se almacenan al registrar un producto al carrito. Elaborado por: Los autores

En la Figura 36, la función “getCart”, permite obtener los datos almacenados en el array cart, los datos que se obtienen son referentes al usuario que esté solicitando dicha información.

Figura 36

Obtener productos almacenados en el array

```
80    static getCart(userId) {
81        var products = cart.filter(function (userProduct) { return userProduct.idUser == userId; });
82        return products;
83    }
```

Nota: En la siguiente figura se puede observar la función para obtener los productos agregados al carrito. Elaborado por: Los autores

3.2.7. Pantalla de categoría

La Figura 37 muestra la pantalla de categoría, donde se encuentran diferentes opciones como crear y eliminar categorías, a este módulo solo se puede acceder como usuario administrador.

Figura 37

Panel de administración de categoría

Nota: Ventana donde se muestra las categorías agregadas en el sistema y un formulario para seguir agregando más categorías. Elaborado por: Los autores

Para los demás módulos como permitir la activación de los usuarios con cuentas inactivas, agregar o eliminar atributos en unidad de media, información y presentación, cada uno de estos módulos muestra su propio panel.

La Figura 38 muestra las diferentes funciones para la visualización, ingreso y la eliminación de las categorías.

Figura 38

Funciones que posee el módulo categoría

```
    getAllCategory: async (req, res) => { //Obtenemos todas los categorias
        const category = await pool.query('SELECT * FROM CATEGORIA WHERE CATEGORIA_ESTADO = "ACTIVO"');
        res.render('administrator/category', { category });
    },

    createCategoryPost: async (req, res) => { //Agregamos nueva categoria
        const { CATEGORIA_NOMBRE, CATEGORIA_DESCRIPCION, CATEGORIA_ESTADO } = req.body;
        const newCategory = {
            CATEGORIA_NOMBRE,
            CATEGORIA_DESCRIPCION,
            CATEGORIA_ESTADO
        }
        newCategory.CATEGORIA_NOMBRE = newCategory.CATEGORIA_NOMBRE.toUpperCase();
        newCategory.CATEGORIA_DESCRIPCION = newCategory.CATEGORIA_DESCRIPCION.toUpperCase();
        newCategory.CATEGORIA_ESTADO = 'ACTIVO';
        try {
            await pool.query('INSERT INTO CATEGORIA set ?', [newCategory]);
        } catch (error) {
            return cb(new Error('Error al agregar una nueva categoria'));
        }
        res.redirect('/administrator/category');
    },

    deleteCategory: async (req, res) => { //Eliminamos la categoria seleccionada
        const { id } = req.params;
        await pool.query('UPDATE CATEGORIA SET CATEGORIA_ESTADO = "ELIMINADO" WHERE CATEGORIA.CATEGORIA_ID = ?', [id]);
        res.redirect('/administrator/category');
    },
```

Nota: En la presente imagen se muestra las funciones que tiene el módulo de categoría. Elaborado por: Los autores

3.3.CONTROLES PRINCIPALES

3.3.1. Conexión de bases de datos

En la Figura 39, se define las variables de conexión con sus valores correspondientes para la conexión a la base de datos, en el caso de no existir una conexión o que esta se pierda, mostrará un mensaje informando de los errores.

Figura 39

Conexión de bases de datos

```
pool.getConnection((err, connection) => {
    if (err) {
        if (err.code === 'PROTOCOL_CONNECTION_LOST') {
            console.error('DATABASE CONNECTION WAS CLOSED');
        }
        if (err.code === 'ER_CON_COUNT_ERROR') {
            console.error('DATABASE HAS TO MANY CONNECTIONS');
        }
        if (err.code === 'ECONNREFUSED') {
            console.error('DATABASE CONNECTION WAS REFUSE');
        }
    }

    if (connection) connection.release();
    console.log('DB is Connected');
    return;

});
```

Nota: En la presente figura se puede observar la configuración de la conexión. Elaborado por: Los autores

3.3.2. Creación de sesiones

En la Figura 40, se crea una nueva sesión para el cliente, donde se determina qué datos del objeto del usuario deben almacenarse en la sesión a través de la función (serializeUser). Esto permite guardar la sesión en la base de datos y se utiliza posteriormente para recuperar todo el objeto a través de la función (deserializeUser), de esta manera se determina si un usuario inicio sesión en el sistema.

Figura 40

Creación de sesiones

```
passport.serializeUser((user, done) => {//Esto permite almacenar en sesion
    done(null, user.PERSONA_ID);
});

passport.deserializeUser(async (PERSONA_ID, done) => {
    const rows = await pool.query('SELECT * FROM PERSONA WHERE PERSONA_ID = ?', [PERSONA_ID]);
    done(null, rows[0]);
});
```

Nota: En la presente figura se puede observar la creación de sesiones. Elaborado por: Los autores

3.3.3. Validar ingreso al sistema

En la Figura 41, las funciones “isLoggerdIn”, “isNotLoggedIn” y “isAdmin”, la primera permite detectar si el usuario no inicio sesión y quiere ingresar al módulo de usuario este automáticamente lo envía al módulo de inicio de sesión, en la segunda si el usuario accedió a su cuenta y quiere ingresar al módulo de inicio de sesión este automáticamente le restringe al ingreso del mismo y en la tercera si un usuario no tiene el rol de administrador este le restringe el ingreso al módulo administrador.

Figura 41

Validar ingreso al sistema

```
module.exports = {
    isLoggedIn(req, res, next) {
        if (req.isAuthenticated()) {
            return next();
        }
        return res.redirect('/signin');
    },

    isNotLoggedIn(req, res, next) {
        if (!req.isAuthenticated()) {
            return next();
        }
        return res.redirect('/products');
    },

    isAdmin(req, res, next) {
        try {
            if (req.user.ROL_ID == '1') {
                return next();
            }
            return res.redirect('/products');
        } catch {
            return res.redirect('/signin');
        }
    }

};
```

Nota: En la siguiente figura se muestra cómo se configuran los ingresos a cada uno de los módulos del sistema. Elaborado por: Los autores

3.3.4. Validaciones

En la Figura 42, se observa la función por donde va a pasar cada validación, en donde si existe algún inconveniente se disparará un mensaje de error, en caso contrario el sistema seguirá con su flujo normal.

Figura 42

Función donde pasa todo el flujo de validaciones

```
const yup = require('yup');

function validate(validation) {
    return (req, res, next) => {
        try {
            validation(req.body);
            next();
        } catch (error) {
            next(error);
        }
    };
}
```

Nota: En la presente figura se puede observar la función por donde pasaran todas las validaciones. Elaborado por: Los autores

3.3.5. Ingreso de imágenes

En la Figura 43, se observa la función que carga las imágenes que van a hacer subidas al sistema, en donde si existe algún inconveniente se disparará un mensaje de error, en caso contrario el sistema subirá la imagen.

Figura 43

Ingreso de imágenes al sistema

```
// Agregar imagenes
const storage = multer.diskStorage({
    destination: path.join(__dirname, 'public/img'),
    filename: (req, file, cb, filename) => {
        cb(null, uuidv4() + path.extname(file.originalname));
    }
})

app.use(multer({
    storage,
    fileFilter: (req, file, cb) => {
        if (file.size > 700000) { // Tamaño maximo del archivo(7 Kb)
            return cb(new Error('El peso maximo de la imagen es muy grande, por favor cambie de imagen'));
        } else if (file.mimetype !== "image/png" && file.mimetype !== "image/jpg" && file.mimetype !== "image/jpeg") {
            return cb(new Error('El archivo debe ser una imagen valida'));
        } else {
            return cb(null, true);
        }
    }
}).single('SUBIR_IMAGEN'));

```

Nota: En la presente figura se puede observar la función que permite cargar imágenes al sistema. Elaborado por: Los autores

3.3.6. Envió de correo

En la Figura 44 se muestran las funciones “getTemplate” y “sendEmail”, la primera permite enviar el template o plantilla del mensaje que se enviará al correo del usuario, en la segunda función se envía el asunto.

Figura 44

Envió de correos

```
const getTemplate = (name, link) => {
    return `
      <head>
         <link rel="stylesheet" href="./style.css">
      </head>

      <div id="email___content">
        <h2>Compra Ventas de productos agricolas</h2>
        <h3>Hola ${name}</h3>
        <p>
        Confirmar cuenta,
        <a href="${link}" target="_blank">Ingresa Aquí</a>
        </p>
      </div>
    `;
}

const sendEmail = async (email, subject, html) => {
    try {

        await transporter.sendMail({
            from: `Plataforma web UPS <${mail.user}>`, // sen
            to: email, // list of receivers
            subject, // Subject line
            text: "Validación del token", // plain text body
            html, // html body
        });

    } catch (error) {
        console.log('Algo no va bien con el email', error);
    }
};
```

Nota: En la presente figura se puede observar la función que permite él envió de correos. Elaborado por: Los autores

3.3.7. Token de validación

En la Figura 45 se muestran las funciones “getToken” y “getTokenData”, la primera permite la creación del token que será enviada posteriormente al usuario, en la segunda se valida el token que el usuario posee sea correcto.

Figura 45

Creación y validación del token

```
const jwt = require('jsonwebtoken');

const getToken = (payload) => {
    return jwt.sign({
        data: payload
    },         , { expiresIn: '10m' })// Valido por 10 minutos
}

const getTokenData = (token) => {
    let data = null;
    jwt.verify(token,         , (err, decoded) => {
        if (err) {
            console.log('Error al obtener la data del token');
        } else {
            data = decoded;
        }
    });
    return data;
}
```

Nota: En la presente figura se puede observar la función que permiten la creación y validación del token. Elaborado por: Los autores

CAPÍTULO 4

PRUEBAS Y RESULTADOS

En el presente capítulo se presentará los resultados de las pruebas realizadas del funcionamiento del sistema, con el objetivo de detectar y depurar las fallas, estas pruebas permiten garantizar la calidad del producto y también permiten demostrar que los requerimientos propuestos fueron correctamente desarrollados.

4.1. PRUEBAS DE CAJA NEGRA DE LA PLATAFORMA WEB

Para una mejor apreciación de cómo está funcionando internamente el sistema, se realizó pruebas de caja negra, las cuales se basan en los requerimientos funcionales, verificando que las entradas y salidas sean las esperadas y que las funciones del sistema sean operativas.

Tabla 10

PB1_Iniciar_Sesión_aplicación_web

Caso de prueba	Iniciar sesión
Identificador	PB1_Iniciar_Sesión_aplicación_web
Función a probar	Inicio de sesión de la plataforma web
Objetivo	Iniciar sesión en la aplicación web
Descripción	Verificar si se cumple los requisitos satisfactoriamente al iniciar sesión e identificar si tiene un tratamiento de errores al momento de insertar datos incorrectos
Precondiciones	El usuario debe estar registrado previamente en el sistema y tener sus datos guardados en la base.
Resultados esperados	Permitir el acceso al panel de usuario. Desplegar los mensajes con los errores de cada campo.

Número de prueba	Acción de usuario	Resultados
1	Usuario ingresa su correo y contraseña correcta	Accede sin inconvenientes al panel de usuario.
2	Usuario ingresa mal su correo e ingresa su contraseña correcta	El sistema despliega un mensaje de que el correo es inexistente.
3	Usuario ingresa bien su correo e ingresa mal su contraseña	El sistema despliega un mensaje de que su contraseña es incorrecta.

Nota: La tabla muestra las pruebas realizadas en el inicio de sesión y sus respectivos resultados.

Elaborado por: Los autores

Tabla 11

PB2_Crear_Cuenta

Caso de prueba	Crear cuenta
Identificador	PB2_Crear_Cuenta
Función a probar	Creación de cuenta
Objetivo	Crear una cuenta en la aplicación web
Descripción	Verificar si se cumple los requisitos satisfactoriamente al crear una cuenta e identificar si tiene un tratamiento de errores al momento de insertar datos incorrectos.
Precondiciones	
Resultados esperados	Permitir el registro del usuario. Verificación campo solicitado. Desplegar los mensajes con los errores de cada campo.

Número de prueba	Acción de usuario	Resultados
1	Completar todos los campos solicitados en el formulario	Se genera una nueva cuenta.
2	No completar al menos un campo	El sistema despliega un mensaje notificando que no se permiten campos vacíos.
3	Introducir un correo que ya exista en el sistema	El sistema despliega un mensaje que notifica que el correo ya se encuentra registrado.

Nota: La tabla muestra las pruebas realizadas en la creación de una cuenta y sus respectivos resultados. Elaborado por: Los autores

Tabla 12

PB3_Ingresar_Producto

Caso de prueba	Ingresar nuevos productos al sistema
Identificador	PB3_Ingresar_Productos
Función probar	Ingresar un nuevo producto en el sistema
Objetivo	Probar el funcionamiento de la aplicación web al ingresar un producto
Descripción	Verificar si se cumple los requisitos satisfactoriamente al ingresar un nuevo producto e identificar si tiene un tratamiento de errores al momento de insertar datos incorrectos
Precondiciones	El usuario debe iniciar sesión
Resultados esperados	Permitir la inserción de nuevos productos Verificación campo solicitado.

	Desplegar los mensajes con los errores de cada campo.	
Número de prueba	Acción de usuario	Resultados
1	Completar todos los campos solicitados en el formulario	El sistema permite ingresar un nuevo producto.
2	No completar al menos un campo	El sistema despliega mensaje de error que comunica que esos campos son obligatorios
3	Introducir un archivo que no sea una imagen	El sistema despliega un mensaje que notifica que solo se aceptan imágenes

Nota: La tabla muestra las pruebas realizadas en el ingreso de un producto en el sistema y sus respectivos resultados. Elaborado por: Los autores

Tabla 13

PB4_Realizar_Pedido

Caso de prueba	Realizar pedido
Identificador	PB4_Realizar_Pedido
Función a probar	Ejecutar una compra a través de la aplicación web
Objetivo	Ejecutar una compra para comprobar su funcionamiento
Descripción	Verificar si se cumple los requisitos satisfactoriamente al realizar un pedido
Precondiciones	El usuario debe haber iniciado sesión
Resultados esperados	Presentar la interfaz de carrito de compras con los respectivos productos agregados. Permitir realizar un pedido.

	Desplegar las notificaciones de error.	
Número de prueba	Acción de usuario	Resultados
1	Agregar productos al carrito	Se agrega los productos seleccionados al carrito
2	Realizar pedido	Realiza los cálculos relacionados con los productos escogidos y muestra mensaje de compra realizada con éxito
3	Cancelar pedido	Se elimina los productos agregados al carrito y no realiza el pedido

Nota: La tabla muestra las pruebas realizadas al momento de realizar un pedido y sus respectivos resultados. Elaborado por: Los autores

Tabla 14

PB5_Ingreso_Modulo_Administrador

Caso de prueba	Ingresar al módulo administrador sin permisos
Identificador caso de prueba	PB5_Ingreso_Modulo_Administrador
Función probar	Ingresar al módulo administrador sin tener el perfil de administrador
Objetivo	Probar el funcionamiento del sistema al momento de ingresar al módulo administrador sin tener permisos.
Descripción	Verificar si se cumple los requisitos satisfactoriamente al ingresar al módulo de administrador y ver si tiene un tratamiento de validaciones al momento de ingresar sin permisos de administrador
Precondiciones	El usuario debe haber iniciado sesión

Resultados esperados	Denegar el acceso al módulo administrador	
Número de prueba	Acción de Usuario	Resultados
1	Prueba de ingreso hacia el módulo administrador	Deniega el acceso al mismo, redirigiendo al panel de usuario.

Nota: La tabla muestra las pruebas realizadas al ingresar al módulo administrador sin tener permisos del mismo y sus respectivos resultados. Elaborado por: Los autores

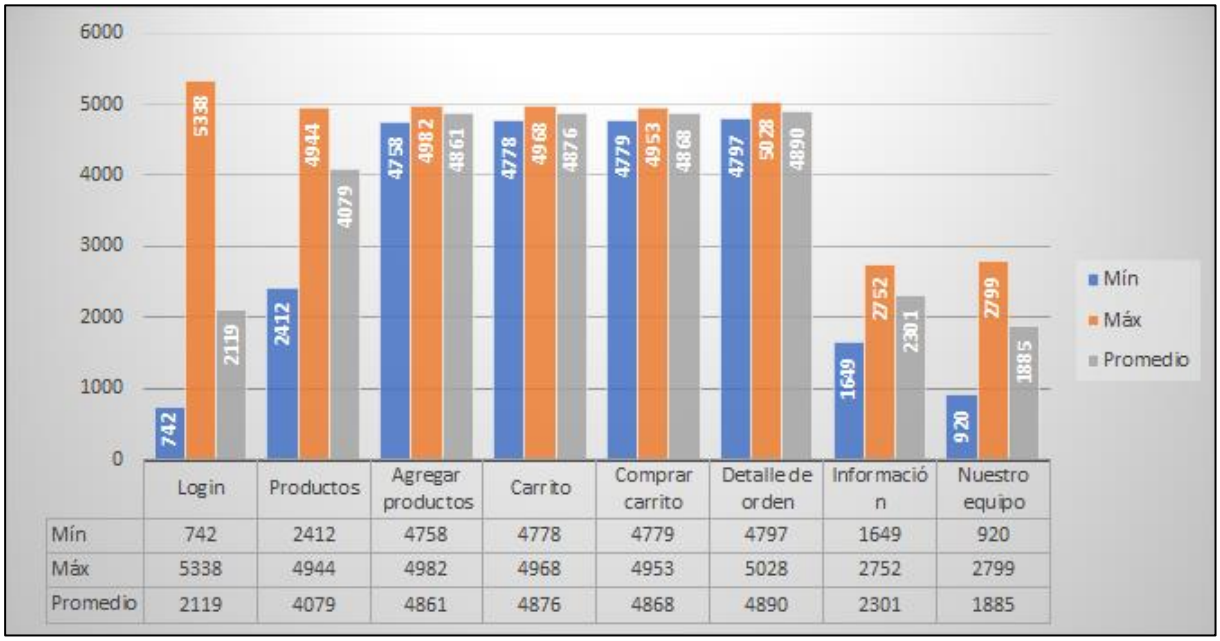
4.2. PRUEBAS DE CARGA

Para las pruebas de carga se utilizó el software JMeter, estas pruebas se realizaron en el hosting gratuito Heroku, se configuro un servidor Proxy Http para cada una de las diferentes peticiones de la plataforma web, estas pruebas fueron realizadas en el navegado Mozilla Firefox, las pruebas fueron evaluadas entre 100 usuarios.

En figura 46 se muestran los tiempos de cada una de las peticiones que se genera en la plataforma web, las pruebas se realizaron para 100 usuarios en los que se puede verificar que el tiempo más alto le pertenece a página “Login” con un máximo de 5338 milisegundos y con el tiempo más bajo con 742 milisegundos de esta manera, se obtiene un promedio de 2119 milisegundos. El promedio más alto corresponde a la página “Agregar productos” con un valor de 4861 milisegundos en una interacción de 100 usuarios.

Figura 46

Pruebas de carga 100 usuarios



	Login	Productos	Agregar productos	Carrito	Comprar carrito	Detalle de orden	Información	Nuestro equipo
Mín	742	2412	4758	4778	4779	4797	1649	920
Máx	5338	4944	4982	4968	4953	5028	2752	2799
Promedio	2119	4079	4861	4876	4868	4890	2301	1885

Nota: En la presente figura se puede observar los datos de las pruebas de carga con 100 usuarios en las distintas páginas de la plataforma web. Elaborado por: Los autores

CONCLUSIONES

La creación de la aplicación web fue implementada según las necesidades que poseen los emprendedores o agricultores, para lo cual se implementó varios módulos que permiten al emprendedor gestionar sus productos y al comprador gestionar sus compras de forma fácil, ágil e intuitiva, cumpliendo así el objetivo general planteado.

La incorporación de tecnologías de software libre en el desarrollo del proyecto, permite a los emprendedores y agricultores no estén atados al pago de licencias y otros softwares, reduciendo exponencialmente el costo del desarrollo.

La integración de cada uno de los módulos especificados por el cliente permite que cada usuario pueda vender o adquirir productos, así como el módulo de búsqueda, facilita encontrar los productos, categorías y sector en que se desea realizar un pedido de productos de forma rápida y ágil.

La interacción de los usuarios de la plataforma es importante para permitir una amplía opción a la hora de adquirir un producto en específico, de tal manera que la plataforma está diseñada para gestionar nuevas categorías y no limitar la lista de opciones de productos o emprendimientos que se puede encontrar en la plataforma, como también tener un control de aquellos usuarios que no interactúe de manera recurrente o de usuarios que no sean verificados de esta manera se puede encontrar información de productos actualizados y productores disponibles y satisfacer las necesidades de los compradores.

RECOMENDACIONES

Se recomienda que, se agregue un módulo de pago en línea, el cual ayudará a ser más eficientes y competitivos, garantizando seguridad a los consumidores.

Para garantizar que la plataforma web funcione de forma continua, es recomendable que se realicen mantenimientos al sistema periódicamente, permitiendo que la aplicación siga mejorando y en caso de tener errores, corregirlos, esto con el fin de asegurar que la web tenga un buen rendimiento con el tiempo.

Para un mejor control de inventario se puede incorporar un módulo estadístico, en el que cada uno de los productores pueda verificar la cantidad de productos que vende cada mes y cada año, esto permitirá identificar que productos poseen una alta demanda en un mes o temporada determinada.

La integración de un módulo de envíos que permita a los usuarios recibir su pedido en una fecha determinada, de esta manera se puede ampliar la plataforma para brindar un servicio completo y personalizado que atraerá a futuros productores y compradores a registrase a la plataforma web.

LISTA DE REFERENCIAS

Paginas Web

Neosoft . (08 de Enero de 2018). *Neosoft* . Obtenido de https://www.neosoft.es/blog/que-es-una-aplicacion-web/

Atlassian. (17 de Julio de 2019). *Atlassian*. Obtenido de https://www.atlassian.com/es/git/tutorials/what-is-version-control

Canive, T., & Balet, R. (27 de Septiembre de 2020). *Sinnaps*. Obtenido de https://www.sinnaps.com/blog-gestion-proyectos/metodologia-xp

Ceballos, F. J. (2004). *unam*. Obtenido de https://programas.cuaed.unam.mx/repositorio/moodle/pluginfile.php/1023/mod_resource/content/1/contenido/index.html

deyde. (16 de Diciembre de 2020). *deyde.* Obtenido de https://deyde.com/news/que-es-un-gestor-de-datos-y-sus-funciones/

Drauta. (11 de Febrero de 2020). *Drauta*. Obtenido de https://www.drauta.com/5-softwares-de-control-de-versiones

emfasi. (02 de Octubre de 2014). *emfasi*. Obtenido de https://www.emfasi.com/desarrollo-de-aplicaciones-web

kinsta. (13 de Mayo de 2021). *kinsta.* Obtenido de https://kinsta.com/es/base-de-conocimiento/que-es-node-js/

notatecnologica. (16 de Diciembre de 2020). *notatecnologica*. Obtenido de https://notatecnologica.com/programas/arquitectura-cliente-servidor/

Ramos, R. (13 de Mayo de 2020). *Rafa Ramos*. Obtenido de https://soyrafaramos.com/que-es-javascript-para-que-sirve/

reactiveprogramming. (203 de Diciembre de 2020). *reactiveprogramming*. Obtenido de https://reactiveprogramming.io/blog/es/estilos-arquitectonicos/cliente-servidor

Robledano, Á. (24 de Septiembre de 2019). *Open Webinars*. Obtenido de https://.net/blog/que-es-mysql/

rockcontent. (12 de Abril de 2020). *rockcontent*. Obtenido de https://rockcontent.com/es/blog/bootstrap/

sinnaps. (27 de Septiembre de 2020). *sinnaps*. Obtenido de https://www.sinnaps.com/blog-gestion-proyectos/metodologia-xp

Visual Studio Code. (29 de Abril de 2015). *Visual Studio Code*. Obtenido de https://code.visualstudio.com/docs

Yanina, M. (04 de Septiembre de 2019). *openwebinars*. Obtenido de https://openwebinars.net/blog/que-es-node-package-manager/